HTTP API Reference

Online Barcode Generator

Table of Contents

Chapter 1. Introduction

The Online Barcode Generator includes an HTTP API that is only accessible with a valid license key.

The license key will be provided by Terry Burton Consulting Ltd, or an authorised agent, and must be passed in the `lic_key` parameter of all HTTP requests made to the API, as described below.

HTTP API Reference

The HTTP API is a stateless, request-response interface that can be called via GET or POST requests to create barcode images:

- For GET requests the data is provided using query parameters.
- For POST requests the data can be provided in the request body as form-encoded data.

Chapter 2. `imagegen` endpoint

The `imagegen` endpoint is used to generate barcode images.

Request parameters:

`action`

One of:

- `eps`: Response in Encapsulated PostScript (EPS) format.
- `pdf`: Response in Portable Document Format (PDF).
- `svg`: Response in Scalable Vector Graphics (SVG) format.
- `png`: Response in 8-bit colormap, non-interlaced Portable Network Graphics (PNG) format.
- `jpg`: Response in 8-bit gray/sRGB Joint Photographic Experts Group (JPEG) format.
- `info`: Upon success, a JSON formatted response containing:
  - `success`: “1” indicates success. “0” indicates an error. In the unsuccessful case, the error will be indicated by:
    - `errormsg`: User-friendly error message. (Strings subject to future change.)
    - `errorcode`: Machine-friendly error code. (Strings should be relatively stable, which may be useful for translation purposes.)
  - `pngb64`: Image as a base64-encoded PNG. Present when `preview=png` (the default). For inline use in a `src` attribute of an HTML img tag, the data can be prefixed with “data:image/png;base64,”.
  - `svgb64`: Image as a base64-encoded SVG. Present when `preview=svg`. For inline use in a `src` attribute of an HTML img tag, the data can be prefixed with “data:image/svg+xml;base64,”.
  - `height`: Height of the image in pixels.
  - `width`: Width of the image in pixels.
  - The recipient should be tolerant of any additional fields, which may be added in future.

`encoder`
: The name of the encoder for the required barcode symbology. See the Symbologies Reference.

`data`
: The barcode message data. See the Symbologies Reference.

`options`
: Options affecting the format and rendering of the barcode. See the Symbologies Reference. *(Optional)*

  In addition to the options specified in the Symbologies Reference, the following specialised options are defined:

  - `crop`: Trim the quiet zone that surrounds the printed barcode elements.
  - `unpackaged`: Do not compress the BWIPP routines within the EPS file output.

`scale_x`
: Horizontal scale factor. Must be greater than 0 and at most 15. *(Optional, default: 2)*

`scale_y`
: Vertical scale factor. Must be greater than 0 and at most 15. *(Optional, default: 2)*

`rotate`
: Rotation factor in degrees. Must be between -360 and 360. *(Optional, default: 0)*

`preview`
: For `action=info`, specifies which image format to embed in the JSON response. *(Optional)*

  - `png`: Return `pngb64` field with base64-encoded PNG. *(Default)*
  - `svg`: Return `svgb64` field with base64-encoded SVG.

`lic_key`
: The license key. *(Required if licenses have been enabled)*

`csrf_token`
: A CSRF token to prevent unauthorised, direct access to the API. *(Required if CSRF tokens have been enabled and a valid `lic_key` is not provided.)*

GET example

A GET request passes all parameters in the query string. Directly browsing to the following URL will produce a PNG image:

```
http://<ENDPOINT_URL>/imagegen?action=png&encoder=qrcode&data=Hello&options=ec
```

```
level%3DH
```

NOTE When using GET requests, parameter values must be percent-encoded. This is especially important for the `options` parameter, which contains `=` characters that would otherwise be interpreted as parameter delimiters. See URL encoding below.

Using `curl`:

```
curl -G 'http://<ENDPOINT_URL>/imagegen' \
  -o barcode.png \
  --data-urlencode 'action=png' \
  --data-urlencode 'encoder=ean13' \
  --data-urlencode 'data=123456789012 12345' \
  --data-urlencode 'options=includetext guardwhitespace' \
  --data-urlencode 'scale_x=5' \
  --data-urlencode 'scale_y=5'
```

POST example

A POST request passes parameters in the request body as form-encoded data. This avoids URL encoding issues and keeps parameters out of server access logs.

```
curl 'http://<ENDPOINT_URL>/imagegen' \
  -o barcode.png \
  -d 'action=png' \
  -d 'encoder=ean13' \
  -d 'data=123456789012 12345' \
  -d 'options=includetext guardwhitespace' \
  -d 'scale_x=5' \
  -d 'scale_y=5'
```

TIP We recommend the use of POST when the request includes a `lic_key` parameter, as the license key will not appear in URLs, browser history, or server access logs.

URL encoding

When making GET requests, all parameter values must be percent-encoded. The `options` parameter is a common source of errors because its value contains `=` signs (e.g. `eclevel=H`) and spaces, both of which have special meaning in a query string.

Correct GET URL for `options=includetext eclevel=H`:

```
/imagegen?action=png&encoder=qrcode&data=Hello&options=includetext%20eclevel%3
DH
```

Common mistakes:

Incorrect URL	Problem
`&options=includetext eclevel=H`	The `=` in `eclevel=H` is parsed as a new parameter named `eclevel` with value `H`, rather than as part of the `options` value.
`&options=includetext+eclevel%3DH`	The `+` is only treated as a space in form-encoded POST bodies. In a URL query string, `+` is ambiguous — use `%20` instead.
`&options=includetext%20eclevel=H`	The space is correctly encoded but the `=` is still unescaped. Both special characters must be encoded.

POST requests using `application/x-www-form-urlencoded` (the default for `curl -d` and HTML forms) handle encoding automatically, which is another reason to prefer POST.

Chapter 3. `healthcheck` endpoint

The `healthcheck` endpoint returns HTTP/200 if the service has successfully loaded.

There are no request parameters.

Chapter 4. Error codes

Errors are returned as JSON with the following structure:

```
{"success":"0", "errorcode":"<code>", "errormsg":"<message>"}
```

Error codes are prefixed by category. The `errorcode` field is intended for programmatic use. The `errormsg` field is human-readable and may change. New error codes may be added in future releases; clients should handle unknown codes gracefully based on their prefix.

4.1. Barcode errors (`bwipp.*`)

These indicate invalid input to the barcode encoder. The error code format is `bwipp.<encoder>.<error>`, for example `bwipp.ean13.badLength`. Refer to the Symbologies Reference for encoder-specific data requirements.

4.2. API errors (`api.*`)

These indicate problems with the request parameters or access control. Most `api.*` errors include a descriptive `errormsg` that indicates what to correct (e.g. `api.ActionUnknown`, `api.ScaleOutOfRange`, `api.EncoderUnknown`).

Some notable examples:

Error code	Description and action
`api.canvasTooLarge`	The barcode output dimensions exceed the configured `MAX_CANVAS` limit. Reduce the `scale_x` / `scale_y` values, or ask the service operator to increase `MAX_CANVAS`.
`api.rateLimitExceeded`	Too many requests in the current time window. The response includes HTTP 429 and a `Retry-After` header — wait the indicated number of seconds before retrying.
`api.licenseUsedUp`	The lifetime request limit for this license has been reached (HTTP 429). Contact the service operator for a new or expanded license.
`api.licenseKeyExpired`	The license key is outside its validity window or has been revoked. Request a renewed license from the service operator.
`api.invalidLicenseKeyToken`	The license key signature is invalid. The key may have been corrupted. Verify the key was copied correctly; if the problem persists, request a new license from the service operator.
`api.sessionTimeout`	The web session has expired. Resubmit the request — the web application will generate a fresh session automatically. If this occurs frequently, the service operator can increase `SESSION_TIMEOUT`.

4.3. PostScript errors (`ps.*`)

`ps.general` indicates that an error occurred in the PostScript interpreter that is not a barcode input error. This is uncommon. Retry the request; if the problem persists, report it to the service operator, who can check the container logs for details.

4.4. Internal errors (`internal.*`)

These indicate system-level failures that cannot be resolved by the API user. The response includes a `correlation_id` field for diagnosis. Retry the request; if the problem persists, report the `correlation_id` to the service operator.

Some examples:

Error code	Meaning
`internal.requestTimeout`	The overall request time budget was exceeded. The rendering workers may be at capacity.
`internal.workerUnavailable`	A rendering worker could not be reached or timed out.
`internal.redisError`	The Redis cache service is unavailable.

Report the `correlation_id` to the service operator for diagnosis.

Chapter 5. Symbologies Reference

5.1. Point of Sale

5.1.1. EAN-13

EAN-13 is an extension of the UPC-A barcode symbology that usually carries a GTIN-13. It was designed by the International Article Numbering Association in 1976 for identification of retail goods at point of sale outside of the US.

Also known as: EAN, European Article Number, International Article Number, JAN, JAN-13, IAN, WPC, SAAN, UCCET, ABAC, BCCI, ICA, MANA, KANC, ANA, ANC.

Misnomers: UCC-13, GTIN-13

Variants:

- EAN-13+2 is an extension of EAN-13 that includes a two-digit add-on.
- EAN-13+5 is an extension of EAN-13 that includes a five-digit add-on.
- EAN-99 is a special form of EAN-13 starting with `99` that is used as an in-store coupon.
- EAN-8 is a barcode symbology derived from EAN-13 that is designed for small packaging. It uses a distinct numbering system based on GTIN-8.
- ISBN is a variant of EAN-13 used to identify books.
- ISMN is a variant of EAN-13 used to identify printed music.
- ISSN is a variant of EAN-13 used to identify periodicals.
- EAN-13 Composite is a variant of EAN-13 that should be used when a CC-A or CC-B GS1 Composite 2D component is required.

Standards: ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data field for a EAN-13 may contain twelve or thirteen digits, optionally followed by a space then two or five digits if an EAN-2 or EAN-5 add-on is required.
- If twelve digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.
- The **includetext** option should normally be supplied.
- The **guardwhitespace** option enables the display of whitespace guard marks.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    9520123456788
Options: includetext guardwhitespace
Encoder: ean13
```

```
Data:    952012345678
Options: includetext guardwhitespace
Encoder: ean13
```

9 520123 456788 >

A symbol that includes a five-digit add-on:

```
Data:    952012345678 54499
Options: includetext guardwhitespace
Encoder: ean13
```

9 520123 456788 54499 >

5.1.2. EAN-8

EAN-8 is derived from the EAN-13 barcode symbology and is designed for small packaging. It usually carries a GTIN-8.

Also known as: JAN-8.

Misnomers: UCC-8.

Variants:

- EAN-Velocity is a special form of EAN-8 starting with `0` that is used for in-store coupons.
- EAN-13 is a longer variant of EAN-8 which has a distinct number system based on GTIN-13.
- EAN-8 Composite is a variant of EAN-8 that should be used when a CC-A or CC-B GS1 composite 2D component is required.
- EAN-8+2 is a **non-standard** extension of EAN-8 that includes a two-digit add-on.
- EAN-8+5 is a **non-standard** extension of EAN-8 that includes a five-digit add-on.
- Marks & Spencer is a derivative of EAN-8 used by the UK retailer.

Standards: ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data field takes either seven or eight digits.
- When the **permitaddon** option is provided, the initial seven or eight digits can be followed by a space then two or five digits where a EAN-2 or EAN-5 add-on is required. It should be noted that **the use of a 2-digit or 5-digit add-on with EAN-8 is not supported by any symbology standard**.
- If seven digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.
- The **includetext** option should normally be supplied.
- The **guardwhitespace** option enables the display of whitespace guard marks.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    95200002
Options: includetext
Encoder: ean8
```

```
Data:    9520000
Options: includetext
Encoder: ean8
```

9520 0002

Truncated with whitespace guards:

```
Data:    95200002
Options: includetext height=0.5 guardwhitespace
Encoder: ean8
```

< 9520 0002 >

5.1.3. UPC-A

The **UPC-A** barcode symbology is used for identification of retail goods at point of sale inside of the US. It usually carries a GTIN-12.

Also known as: UPC, Universal Product Code.

Misnomers: UCC-12.

Variants:

- UPC-A+2 is an extension of UPC-A that includes a two-digit add-on.
- UPC-A+5 is an extension of UPC-A that includes a five-digit add-on.
- UPC-E is a barcode symbology derived from UPC-A that is designed for small packaging.
- UPC-A Composite is a variant of UPC-A that should be used when a CC-A or CC-B GS1 composite 2D component is required.
- A UPC-A symbol can be converted to an EAN-13 symbol by prefixing the GTIN-12 with 0 to make the equivalent GTIN-13.

Standards: ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data field for a UPC-A may contain eleven or twelve digits, optionally followed by a space then two or five digits if an EAN-2 or EAN-5 add-on is required.
- Alternatively, the data field may contain seven or eight digits of a UPC-E to produce the equivalent UPC-A symbol.
- If eleven digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.
- The **includetext** option should normally be supplied.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    012345000058
Options: includetext
Encoder: upca
```

```
Data:    01234500005
Options: includetext
Encoder: upca
```

0 12345 00005 8

A symbol that includes a five-digit add-on:

```
Data:    012345000058 54499
Options: includetext guardwhitespace
Encoder: upca
```

54499>

0 12345 00005 8

5.1.4. UPC-E

UPC-E is a compacted form of the UPC-A barcode symbology that usually carries a GTIN-12 that has been zero compressed.

Also known as: UPC-E0.

Variants:

- UPC-E+2 is an extension of UPC-E that includes a two-digit add-on.
- UPC-E+5 is an extension of UPC-E that includes a five-digit add-on.
- UPC-A is the full size form of UPC-E.
- UPC-E Composite is a variant of UPC-E that should be used when a CC-A or CC-B GS1 Composite 2D component is required.
- UPC-E1 is a UPC-E with a number system of *1*. This is not defined by any symbology standard.

Standards: ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data field takes either seven or eight digits, optionally followed by a space then two or five digits if an EAN-2 or EAN-5 add-on is required.
- Alternatively, the data field may contain eleven or twelve digits of a UPC-A to produce the equivalent UPC-E symbol, provided that the input can be zero suppressed.
- If seven digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.
- For a regular UPC-E the primary data should start with "0". For a UPC-E1 the data should start with "1".
- The **includetext** option should normally be supplied.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    01234558
Options: includetext
Encoder: upce
```

```
Data:    0123455
Options: includetext
Encoder: upce
```

0 123455 8

A truncated symbol:

```
Data:    01234558
Options: includetext height=0.5
Encoder: upce
```

0 123455 8

5.1.5. ISBN

An **ISBN** barcode is a variant of EAN-13 that is used to identify books.

Also known as: ISBN-13, International Standard Book Number, Bookland EAN, Bookland EAN-13.

Variants:

- ISBN-10 is a legacy format that was deprecated for public use after 1st January 2007.

Standards: ISO 2108, ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data should contain twelve or thirteen digits separated appropriately by dash characters -.
- The data can also be provided in legacy ISBN-10 format as nine or ten digits separated appropriately by dash characters -. This will be automatically upgraded to the ISBN-13 format.
- If the last digit of the primary data is not given then the ISBN check digit is calculated automatically, otherwise it will be verified.
- The primary data can optionally be followed by a space then two or five digits if an EAN-2 or EAN-5 add-on is required.
    - A five-digit add-on is typically used to represent the price supplicant.
    - A two-digit add-on is not typically used with ISBN.
- The **includetext** option should normally be supplied.

- The **guardwhitespace** option enables the display of whitespace guard marks.
- The following options are also relevant to this barcode symbology:
  - **isbntextfont**: PostScript font name for text above symbol
  - **isbntextsize**: Font size for the text above symbol, in points
  - **isbntextxoffset**: Horizontal position of ISBN text, in points
  - **isbntextyoffset**: Vertical position of ISBN text, in points
- *Deprecated.* The **legacy** option prevents ISBN-10 input from being upgraded to ISBN-13 and will result in a symbol that is obsolete and should not be used at point of sale.

Examples

ISBN-13

Identical symbols, input provided with and without an ISBN check digit:

```
Data:    978-1-873671-00-9
Options: includetext
Encoder: isbn
```

```
Data:    978-1-873671-00
Options: includetext
Encoder: isbn
```

ISBN 978-1-873671-00-9

9 781873 671009

An ISBN with a five-digit add-on:

```
Data:    978-1-873671-00-9 54499
Options: includetext guardwhitespace
Encoder: isbn
```

ISBN 978-1-873671-00-9

9 781873 671009 54499>

The following ISBN-10 input will be automatically upgraded to a valid ISBN-13 symbol:

```
Data:    1-86074-271-8
Options: includetext
Encoder: isbn
```

```
Data:    1-86074-271
Options: includetext
Encoder: isbn
```

ISBN 978-1-86074-271-2

9 781860 742712

ISBN-10 (Legacy)

Note that ISBN-10 is a legacy format not for use at P.O.S.

The following will generate an obsolete ISBN-10 symbol:

```
Data:    1-86074-271-8
Options: legacy includetext guardwhitespace
Encoder: isbn
```

```
Data:    1-86074-271
Options: legacy includetext guardwhitespace
Encoder: isbn
```

ISBN 1-86074-271-8

9 781860 742712 >

5.1.6. ISMN

An **ISMN** barcode is a variant of EAN-13 with a prefix *979* that is used to identify printed music.

Also known as: International Standard Music Number, ISMN-13, Musicland EAN-13.

Variants:

- ISMN-10 is a legacy format that was deprecated for public use.

Standards: ISO 10957, ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data should contain twelve or thirteen digits separated appropriately by dash characters -.
- The data can also be provided in legacy ISMN-10 format start *M-* then eight or nine digits separated appropriately by dash characters -. This will be automatically upgraded to the ISMN-13 format.
- If the last digit of the primary data is not given then the ISMN check digit is calculated automatically, otherwise it will be verified.
- The primary data can optionally be followed by a space then two or five digits if an EAN-2 or EAN-5 add-on is required.
- The **includetext** option should normally be supplied.
- The **guardwhitespace** option enables the display of whitespace guard marks.
- The following options are also relevant to this barcode symbology:
  - **ismntextfont**: PostScript font name for text above symbol
  - **ismntextsize**: Font size for the text above symbol, in points
  - **ismntextxoffset**: Horizontal position of ISMN text, in points
  - **ismntextyoffset**: Vertical position of ISMN text, in points
- *Deprecated.* The **legacy** option prevents ISMN-10 input from being upgraded to ISMN-13 and will result in a symbol that is obsolete and should not be used at point of sale.

Examples

ISMN-13

Identical symbols, input provided with and without an ISMN check digit:

```
Data:    979-0-2600-0043-8
Options: includetext
Encoder: ismn
```

```
Data:    979-0-2600-0043
Options: includetext
```

```
Encoder: ismn
```

ISMN 979-0-2600-0043-8

9 790260 000438

The following ISMN-10 input will be automatically upgraded to a valid ISMN-13 symbol:

```
Data:    M-345-24680-5
Options: includetext
Encoder: ismn
```

```
Data:    M-345-24680
Options: includetext
Encoder: ismn
```

ISMN 979-0-345-24680-5

9 790345 246805

ISMN-10 (Legacy)

Note that ISMN-10 is a legacy format not for use at P.O.S.

The following will generate an obsolete ISMN-10 symbol:

```
Data:    M-345-24680-5
Options: legacy includetext guardwhitespace
Encoder: ismn
```

```
Data:    M-345-24680
Options: legacy includetext guardwhitespace
Encoder: ismn
```

ISMN M-345-24680-5

9 790345 246805 >

5.1.7. ISSN

An **ISSN** barcode is an EAN-13 with prefix *977* used to identify periodicals.

Also known as: International Standard Serial Number.

Standards: ISO 3297, ISO/IEC 15420, BS EN 797, GS1 General Specifications.

Data and Options

- The data should contain the seven or eight digits ISSN separated by a dash characters -, followed by a two-digit sequence variant, optionally followed by two or five digits if a two-digit add-on or five-digit add-on is required.
    - A two-digit add on is typically used to represent the issue number.
    - A five-digit add on is not typically used with ISSN.
- If the last digit of the ISSN data is not given then the ISSN check digit is calculated automatically, otherwise it will be verified.
- The **includetext** option should normally be supplied.
- The **guardwhitespace** option enables the display of whitespace guard marks.
- The following options are also relevant to this barcode symbology:
    - **issntextfont**: PostScript font name for text above symbol
    - **issntextsize**: Font size for the text above symbol, in points
    - **issntextxoffset**: Horizontal position of ISSN text, in points
    - **issntextyoffset**: Vertical position of ISSN text, in points

A sequence variant is a two-digit number that usually starts at zero and is incremented whenever the recommended retail price is amended, where applicable.

Examples

Identical symbols, input provided with and without an ISSN check digit and having sequence number *00*:

```
Data:    0317-8471 00
Options: includetext guardwhitespace
```

```
Encoder: issn
```

```
Data:    0317-847 00
Options: includetext guardwhitespace
Encoder: issn
```

ISSN 0317-8471

9 770317 847001 >

An ISSN with sequence number *03* and a two-digit add-on representing issue number *17*:

```
Data:    0317-8471 03 17
Options: includetext guardwhitespace
Encoder: issn
```

```
Data:    0317-847 03 17
Options: includetext guardwhitespace
Encoder: issn
```

ISSN 0317-8471

9 770317 847032 17 >

5.2. Two-Dimensional

5.2.1. Aztec Code

Aztec Code is a 2D matrix-style barcode symbology. It can encode full 256-character extended-ASCII.

Variants:

- Compact Aztec Code is a reduced form of Aztec Code used in applications where the space for the symbol is restricted.
- Aztec Runes are a set of small barcode symbols that are used for special applications.

Standards: ISO/IEC 24778, ANSI/AIM BC13 - ISS Aztec Code.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^ECI000000` to `^ECI999999`: ECI indicators
- The **eclevel** option is used to specify the percentage of error correction to be applied when expanding the data, by default `23`.
- The **ecaddchars** option is used to specify how many additional error correction characters to apply the data once expanded by the eclevel percentage, by default `3`.
- The **layers** option is used to specify a particular number of layers in which to encode the data, between `1` and `32`. By default the encoder will create a symbol with be minimal number of layers to encode the given data.
- The **readerinit** option denotes that the symbol is used for programming the barcode reader. Reader programming is only compatible with 1- to 22-layer full-range Aztec Code symbols.
- The **raw** option denotes that the data field is providing the input as a pre-encoded bitstream suitable for direct low-level encoding.
- *Deprecated: Use Compact Aztec Code instead.* The **format** option can also be used to create Compact Aztec Code symbols, using `format=compact`.
- *Deprecated: Use Aztec Runes instead.* The **format** option can also be used to create Aztec Code "runes", using `format=rune`. In this case the rune symbol number should be given in the data field.

Examples

A symbol with a fixed number of layers:

```
Data:    ABC123
Options: layers=3
Encoder: azteccode
```

A compact symbol using raw bitstream input:

```
Data:    00100111001000000101001101111000010100111100101000000110
Options: raw format=compact
Encoder: azteccode
```

A symbol with additional error correction characters:

```
Data:    TEST
Options: ecaddchars=20 layers=4
Encoder: azteccode
```

5.2.2. Compact Aztec Code

Compact Aztec Code is a reduced form of the Aztec Code barcode that is used in applications where the space for the symbol is restricted.

Variants:

- Aztec Code is the larger, more popular variant.
- Aztec Runes are a set of small barcode symbols that are used for special applications.

Standards: ISO/IEC 24778, ANSI/AIM BC13 - ISS Aztec Code.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.

- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^ECI000000` to `^ECI999999`: ECI indicators
- The **eclevel** option is used to specify the percentage of error correction to be applied when expanding the data, by default `23`.
- The **ecaddchars** option is used to specify how many additional error correction characters to apply the data once expanded by the eclevel percentage, by default `3`.
- The **layers** option is used to specify a particular number of layers in which to encode the data, between `1` and `4`. By default the encoder will create a symbol with be minimal number of layers to encode the given data.
- The **readerinit** option denotes that the symbol is used for programming the barcode reader. Reader programming is only compatible with 1-layer Compact Aztec Code symbols.
- The **raw** option denotes that the data field is providing the input as a pre-encoded bitstream suitable for direct low-level encoding.

Examples

A basic symbol:

```
Data:    This is Compact Aztec Code
Options:
Encoder: azteccodecompact
```

A symbol using the parse option with increased error correction:

```
Data:    This is ^065ztec Code
Options: parse eclevel=50 ecaddchars=5
Encoder: azteccodecompact
```

A minimal symbol:

```
Data:    ABC123
```

```
Options:
Encoder: azteccodecompact
```

5.2.3. Aztec Runes

Aztec Runes are a set of small barcode symbols that are used for special applications.

Variants:

Aztec Code is a 2D matrix-style barcode symbology that can encode full 256 character extended-ASCII.

Standards: ISO/IEC 24778, ANSI/AIM BC13 - ISS Aztec Code.

Data and Options

- The data field contains the rune number `0` to `255`.

Examples

A rune encoding the value *25*:

```
Data:    25
Options:
Encoder: aztecrune
```

5.2.4. Data Matrix

The **Data Matrix** symbology is 2D matrix-style barcode that can encode full 256 character extended-ASCII.

Also known as: Data Matrix ECC 200, DataMatrix.

Variants:

- Data Matrix Rectangular is a variant of Data Matrix that uses rectangular symbols.
- Data Matrix Rectangular Extension is a variant of Data Matrix Rectangular that provides a greater range of symbol dimensions.
- GS1 DataMatrix is a variant of Data Matrix that should be used when encoding data that is in GS1 Application Identifier standard format.

- GS1 Digital Link Data Matrix is a variant of Data Matrix that should be used when encoding a GS1 Digital Link URI.
- HIBC Data Matrix is a variant of Data Matrix that should be used when encoding HIBC formatted data.
- Royal Mail Mailmark is a variant of Data Matrix that is used by the Royal Mail postal service on letters and other small mailpieces.

Standards: ISO/IEC 16022, ANSI/AIM BC11 - ISS Data Matrix.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value. This is useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1 DataMatrix for encoding GS1 AI syntax data.*
  - `^PROG`: PROG - Reader programming
  - `^ECI000000` to `^ECI999999`: ECI indicators
- Whenever the data field contains suitable input, the encoder will compress ISO/IEC 15434 structured data (internally using the 05 Macro and 06 Macro codewords.)
- When the **dotty** option is specified the modules are rendered as dots rather than squares. The dot radius can be adjusted using the **inkspread** option.
- The **columns** and **rows** options are used to specify the size of the symbol.
- The **version** option can also be used to specify the symbol size, as `version=RxC`. Valid options are *10x10*, *12x12*, *14x14*, *16x16*, *18x18*, *20x20*, *22x22*, *24x24*, *26x26*, *32x32*, *36x36*, *40x40*, *44x44*, *48x48*, *52x52*, *64x64*, *72x72*, *80x80*, *88x88*, *96x96*, *104x104*, *120x120*, *132x132*, *144x144*
- If **columns**, **rows** and **version** are unspecified the encoder will default to creating a symbol of the specified **format** that is the minimum size to represent the given data.
- The **c40headerlength** option specifies the number of leading characters in the data field that must be stored using C40 encodation.
- The **raw** option denotes that the data field is providing the input as pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.
- *Deprecated*: The **format** option is used to specify the shape of the symbol, either:
  - `square` (default)

  - `rectangle`: Use Data Matrix Rectangular for rectangular symbols.
- *Deprecated: Use Data Matrix Rectangular Extension*. The **dmre** option enables Data Matrix Rectangular Extension symbols based on the ISO/IEC standard, which increases the number of rectangular symbol sizes available.

Examples

Identical symbols, automatically sized, the latter showing use of the parse option:

```
Data:    This is Data Matrix
Options:
Encoder: datamatrix
```

```
Data:    This is ^068ata Matrix
Options: parse
Encoder: datamatrix
```

Identical symbols with a fixed size:

```
Data:    Fixed size
Options: version=48x48
Encoder: datamatrix
```

```
Data:    Fixed size
Options: rows=48 columns=48
Encoder: datamatrix
```

A rectangular symbol with a fixed size:

```
Data:    Rectangular
Options: format=rectangle version=16x48
```

```
Encoder: datamatrix
```

An advanced use containing "hidden data" in the padding codewords as used by some non-standard, "high-security" applications. The technique works by filling the symbol using raw codewords formed from the standard data immediately followed by the non-standard padding data beginning with `^129`:

```
Data:    ^066^067^068^142^052^129^161^056^206^101^251^147
Options: version=16x16 raw
Encoder: datamatrix
```

5.2.5. Data Matrix Rectangular

The **Data Matrix Rectangular** symbology is a rectangular variant of Data Matrix.

Variants:

- Data Matrix is the standard square variant of the symbology.
- Data Matrix Rectangular Extension is a variant of Data Matrix Rectangular that provides a greater range of symbol dimensions.

Standards: ISO/IEC 16022, ANSI/AIM BC11 - ISS Data Matrix.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value. This is useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1 DataMatrix for encoding GS1 AI syntax data.*
  - `^PROG`: PROG - Reader programming
  - `^ECI000000` to `^ECI999999`: ECI indicators
- Whenever the data field contains suitable input, the encoder will compress ISO/IEC 15434 structured data (internally using the 05 Macro and 06 Macro codewords.)

- When the **dotty** option is specified the modules are rendered as dots rather than squares. The dot radius can be adjusted using the **inkspread** option.
- *Deprecated: Use Data Matrix Rectangular Extension*. The **dmre** option enables Data Matrix Rectangular Extension symbols based on the ISO/IEC standard, which increases the number of rectangular symbol sizes available.
- The **columns** and **rows** options are used to specify the size of the symbol.
- The **version** option can also be used to specify the symbol size, as `version=RxC`. Valid options are *8x18*, *8x32*, *12x26*, *12x36*, *16x36*, *16x48*.
- If **columns**, **rows** and **version** are unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data.
- The **raw** option denotes that the data field is providing the input as pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.

Examples

A rectangular symbol with a fixed size:

```
Data:    Rectangular
Options: version=16x48
Encoder: datamatrixrectangular
```

5.2.6. Data Matrix Rectangular Extension

The **Data Matrix Rectangular Extension** symbology is an enhancement to Data Matrix Rectangular that provides a greater range of symbol dimensions.

Also known as: DMRE.

Variants:

- Data Matrix is the standard square variant of the symbology.
- Data Matrix Rectangular is the original rectangular variant of Data Matrix that provides a limited range of symbol dimensions.

Standards: ISO/IEC 21471.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).

- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value. This is useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^PROG`: PROG - Reader programming
  - `^ECI000000` to `^ECI999999`: ECI indicators
- Whenever the data field contains suitable input, the encoder will compress ISO/IEC 15434 structured data (internally using the 05 Macro and 06 Macro codewords.)
- When the **dotty** option is specified the modules are rendered as dots rather than squares. The dot radius can be adjusted using the **inkspread** option.
- The **columns** and **rows** options are used to specify the size of the symbol.
- The **version** option can also be used to specify the symbol size, as `version=RxC`. Valid options are *8x18*, *8x32*, *8x48*, *8x64*, *8x80*, *8x96*, *8x120*, *8x144*, *12x26*, *12x36*, *12x64*, *12x88*, *16x36*, *16x48*, *16x64*, *20x36*, *20x44*, *20x64*, *22x48*, *24x48*, *24x64*, *26x40*, *26x48*, *26x64*.
- If **columns**, **rows** and **version** are unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data.
- The **raw** option denotes that the data field is providing the input as pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.

Examples

A DMRE symbol with a fixed size:

```
Data:    1234
Options: version=8x80
Encoder: datamatrixrectangularextension
```

5.2.7. Han Xin Code

The **Han Xin Code** symbology is a 2D matrix-style barcode symbology that can encode full 256 character extended-ASCII.

Also known as: Chinese Sensible.

Standards: GB/T 21049-2007.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- The **eclevel** option is used to specify the error correction level:
    - `eclevel=L1` - Lowest
    - `eclevel=L2`
    - `eclevel=L3`
    - `eclevel=L4` - Highest
- The **version** option is used to specify the size of the symbol, 1 to 84.
- If unspecified the encoder will select the version of the symbol that is the minimum size to represent the given data at the selected error correction level.

Examples

A fixed-size symbol with maximum error correction:

```
Data:     Han Xin Code
Options: version=10 eclevel=L4
Encoder: hanxin
```

5.2.8. MicroPDF417

The **MicroPDF417** barcode symbology is 2D stacked-linear barcode based on PDF417 that can encode full 256 character extended-ASCII.

Variants:

- PDF417 is a larger variant of the MicroPDF417 barcode.
- HIBC MicroPDF417 is a variant of MicroPDF417 that should be used when encoding HIBC formatted data.

Standards: ISO/IEC 24728, AIM ISS - MicroPDF417.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^ECI000000` to `^ECI811799`: ECI indicators
- Whenever the data field contains suitable input, the encoder will compress ISO/IEC 15434 structured data (internally using the 05 Macro and 06 Macro codewords.)
- The **columns** and **rows** options are used to specify the size of the symbol. Valid values are:
  - *1x11*, *1x14*, *1x17*, *1x20*, *1x24*, *1x28*, *2x8*, *2x11*, *2x14*, *2x17*, *2x20*, *2x23*, *2x26*, *3x6*, *3x8*, *3x10*, *3x12*, *3x15*, *3x20*, *3x26*, *3x32*, *3x38*, *3x44*, *4x4*, *4x6*, *4x8*, *4x10*, *4x12*, *4x15*, *4x20*, *4x26*, *4x32*, *4x38*, *4x44*
- If the **columns** and **rows** are unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data.
- The **rowmult** option is used to specify how tall each bar is, with respect to the minimum module width. The default is 3.
- The **raw** option denotes that the data field is providing the input as a pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.
- *Deprecated: Internal use.* The **cca** option identifies this symbol as a *CC-A* 2D component of a GS1 Composite symbol.
  - Special size rules apply when the **cca** option is given, in which case the **columns** and **rows** options that are used to specify the size of the symbol must be one of:
    - *2x5*, *2x6*, *2x7*, *2x8*, *2x9*, *2x10*, *2x12*, *3x4*, *3x5*, *3x6*, *3x7*, *3x8*, *4x3*, *4x4*, *4x5*, *4x6*, *4x7*
- *Deprecated: Internal use.* The **ccb** option identifies this symbol as a *CC-B* 2D component of a GS1 Composite symbol.

Examples

A basic symbol:

```
Data:    MicroPDF417
Options:
Encoder: micropdf417
```

A symbol using the parse option with fixed dimensions:

```
Data:    MicroP^068F417
Options: parse rows=15 columns=4
Encoder: micropdf417
```

5.2.9. PDF417

The **PDF417** barcode symbology is 2D stacked-linear barcode that can encode full 256 character extended-ASCII.

Variants:

- Compact PDF417 is a shortened form of the PDF417 barcode that is used in applications where the space for the symbol is restricted.
- MicroPDF417 is a smaller variant of the PDF417 barcode.
- HIBC PDF417 is a variant of PDF417 that should be used when encoding HIBC formatted data.

Standards: ISO/IEC 15438, DD ENV 12925, AIM USS - PDF417.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^ECI000000` to `^ECI811799`: ECI indicators.
- The **eclevel** option is used to specify the error correction level, from 1 to 5. The default is to choose a standard level of error correction that is determined by the encoded data length.
- The **fixedeclevel** option will prevent the error correction level from being

opportunistically raised when a better fix to the current matrix is possible.

- The **columns** option specifies the number of columns (or groups of bars) in the output symbol, from 1 to 30.
- The **rows** option specifies the minimum number of rows in the symbol, from 3 to 90.
- If **rows** is unspecified the encoder will select a number that creates a symbol that is the minimum size to represent the given data.
- The **rowmult** option is used to specify how tall each bar is, with respect to the minimum module width. The default is 3.
- The **raw** option denotes that the data field is providing the input as a pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.
- *Deprecated: Use Compact PDF417 instead. The **compact** option is used to create a compact/truncated PDF417 symbol that has fewer bars per row that a standard symbol and hence is more narrow.*
- *Deprecated: Internal use.* The **ccc** option identifies this symbol as a *CC-C* 2D component of a GS1 Composite symbol.

Examples

A basic symbol:

```
Data:    PDF417
Options:
Encoder: pdf417
```

A symbol using the parse option with fixed dimensions:

```
Data:    P^068F417
Options: parse columns=2 rows=15
Encoder: pdf417
```

A symbol with maximum error correction:

```
Data:    Strong error correction
Options: columns=2 eclevel=5
```

```
Encoder: pdf417
```

A symbol using raw codeword input:

```
Data:    ^453^178^121^239
Options: raw columns=2
Encoder: pdf417
```

5.2.10. Compact PDF417

Compact PDF417 is a shortened form of the PDF417 barcode that is used in applications where the space for the symbol is restricted.

Also known as: Truncated PDF417.

Variants:

- PDF417 is the larger, more popular variant.
- MicroPDF417 is a smaller variant of the PDF417 barcode.
- HIBC PDF417 is a variant of PDF417 that should be used when encoding HIBC formatted data.

Standards: ISO/IEC 15438, DD ENV 12925, AIM USS - PDF417.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.

- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
    - `^ECI000000` to `^ECI811799`: ECI indicators
- The **eclevel** option is used to specify the error correction level, from 1 to 5. The default is to choose a standard level of error correction that is determined by the encoded data length.
- The **fixedeclevel** option will prevent the error correction level from being opportunistically raised when a better fix to the current matrix is possible.
- The **columns** option specifies the number of columns (or groups of bars) in the output symbol, from 1 to 30.
- The **rows** option specifies the minimum number of rows in the symbol, from 3 to 90.
- If **rows** is unspecified the encoder will select a number that creates a symbol that is the minimum size to represent the given data.
- The **rowmult** option is used to specify how tall each bar is, with respect to the minimum module width. The default is 3.
- The **raw** option denotes that the data field is providing the input as a pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.

Examples

A compact symbol with four columns:

```
Data:    A truncated PDF417
Options: columns=4
Encoder: pdf417compact
```

5.2.11. QR Code

The **QR Code** symbology is a 2D matrix-style barcode symbology that can encode full 256 character extended-ASCII.

Also known as: Quick Response Code.

Variants:

- Micro QR Code is a small QR Code that is used in applications that require a small symbol space.
- Rectangular Micro QR Code is a rectangular variant that provides a compact symbol suitable for narrow, elongated spaces.

- GS1 QR Code is a variant that should be used when encoding data that is in GS1 Application Identifier standard format.
- GS1 Digital Link QR Code is a variant that should be used when encoding a GS1 Digital Link URI.
- HIBC QR Code is a variant of QR Code that should be used when encoding HIBC formatted data.
- Swiss QR Code is a variant of the QR Code for use with the QR-bill application supporting Swiss payments.

Standards: ISO/IEC 18004, JIS X 0510, ITS - QR Code, AIM ISS - QR Code.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1 QR Code for encoding GS1 AI syntax data.*
  - `^ECI000000` to `^ECI999999`: ECI indicators
- The **eclevel** option is used to specify the error correction level:
  - `eclevel=L` - Low
  - `eclevel=M` - Medium (default)
  - `eclevel=Q` - Quality
  - `eclevel=H` - High
- The **fixedeclevel** option will prevent the error correction level from being opportunistically raised when a better fix to the current matrix is possible.
- The **version** option is used to specify the size of the symbol, 1 to 40.
- If unspecified the encoder will select the version of the symbol that is the minimum size to represent the given data at the selected error correction level.
- *Deprecated*: The **suppresskanjimode** option will prevent the use of Kanji Mode internal encodation. Some readers erroneously consider a Kanji Mode section as an indication that the affected part of the message should be interpreted using a Shift-JIS character set rather than as a data compaction strategy. There's nothing that can be done other than to disable the use of Kanji Mode data compaction for symbols intended to be read by such broken readers.

- *Deprecated*: The **format** option is used to specify the symbol type, either:
  - `full` (default)
  - `micro`: Use Micro QR Code for Micro QR Code symbols.
  - `rmqr`: The standard is still in draft.

Examples

Identical symbols, the latter using the parse option:

```
Data:    QR Code
Options:
Encoder: qrcode
```

```
Data:    QR ^067ode
Options: parse
Encoder: qrcode
```

A fixed-size symbol with increased error correction:

```
Data:    QR CODE 1234
Options: version=10 eclevel=Q
Encoder: qrcode
```

5.2.12. Micro QR Code

The **Micro QR Code** symbology is a smaller variant of QR Code that is used in applications that require a small symbol space.

Also known as: Micro Quick Response Code.

Variants:

- QR Code is the more popular, larger variant.
- Rectangular Micro QR Code is a rectangular variant that provides a compact symbol suitable for narrow, elongated spaces.

Standards: ISO/IEC 18004, JIS X 0510, ITS - QR Code, AIM ISS - QR Code.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- An appropriate size will be selected to work around the following restrictions:
  - An M1 symbol is only compatible with numeric data.
  - An M2 symbol is only compatible with alphanumeric data.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- The **version** option is used to specify the size of the symbol, either `version=M1`, `version=M2`, `version=M3` or `version=M4`.
- The **eclevel** option is used to specify the error correction level:
  - `eclevel=L` - Low (default)
  - `eclevel=M` - Medium; Not compatible with M1 symbols
  - `eclevel=Q` - Quality; Only compatible with M4 symbols
- The **fixedeclevel** option will prevent the error correction level from being opportunistically raised when a better fix to the current matrix is possible.
- If unspecified the encoder will select the version of the symbol that is the minimum size to represent the given data at the selected error correction level.
- *Deprecated*: The **suppresskanjimode** option will prevent the use of Kanji Mode internal encodation. Some readers erroneously consider a Kanji Mode section as an indication that the affected part of the message should be interpreted using a Shift-JIS character set rather than as a data compaction strategy.

Examples

A basic symbol encoding numeric data:

```
Data:    01234567
Options:
```

```
Encoder: microqrcode
```

5.2.13. Rectangular Micro QR Code

The **Rectangular Micro QR Code** (rMQR) symbology is a rectangular variant of QR Code that provides a compact symbol suitable for applications where a narrow, elongated symbol is required, such as labels and small cylindrical items.

Also known as: rMQR.

Variants:

- QR Code is the standard square variant of the symbology.
- Micro QR Code is a smaller square variant.

Standards: ISO/IEC 23941.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1 QR Code for encoding GS1 AI syntax data.*
  - `^ECI000000` to `^ECI999999`: ECI indicators
- The **version** option is **required** and specifies the size of the symbol in `RHxW` format, where H is the height and W is the width in modules:
  - R7x43, R7x59, R7x77, R7x99, R7x139
  - R9x43, R9x59, R9x77, R9x99, R9x139
  - R11x27, R11x43, R11x59, R11x77, R11x99, R11x139
  - R13x27, R13x43, R13x59, R13x77, R13x99, R13x139
  - R15x43, R15x59, R15x77, R15x99, R15x139
  - R17x43, R17x59, R17x77, R17x99, R17x139
- The **eclevel** option is used to specify the error correction level:

  - `eclevel=M` - Medium (default)
  - `eclevel=H` - High
- The **fixedeclevel** option will prevent the error correction level from being opportunistically raised when a better fit to the selected symbol is possible.
- *Deprecated*: The **suppresskanjimode** option will prevent the use of Kanji Mode internal encodation. Some readers erroneously consider a Kanji Mode section as an indication that the affected part of the message should be interpreted using a Shift-JIS character set rather than as a data compaction strategy.

Examples

A numeric-only symbol:

```
Data:    1234
Options: version=R17x139
Encoder: rectangularmicroqrcode
```

A Figure 1 example symbol from the specification:

```
Data:    12345678901234567890123456
Options: version=R13x27 eclevel=M fixedeclevel
Encoder: rectangularmicroqrcode
```

An alphanumeric symbol:

```
Data:    AC-42
Options: version=R7x59 eclevel=H fixedeclevel
Encoder: rectangularmicroqrcode
```

A symbol with ECI:

```
Data:    ^ECI000009^161^162^163^164^165
Options: parse parsefnc version=R17x139 eclevel=M fixedeclevel
```

```
Encoder: rectangularmicroqrcode
```

5.2.14. Swiss QR Code

Swiss QR Code is a variant of the QR Code barcode symbology for use with the QR-bill application supporting Swiss payments.

Standards: Swiss Implementation Guidelines QR-bill, ISO/IEC 18004, ITS - QR Code.

Data and Options

- The data field input is provided in the format described in the Swiss Implementation Guidelines QR-bill document.
- The **version** option is used to specify the size of the symbol, 5 to 25. This is not typically specified.
- If the **version** is unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data at the selected error correction level.
- *Deprecated*: The **suppresskanjimode** option will prevent the use of Kanji Mode internal encodation. Some readers erroneously consider a Kanji Mode section as an indication that the affected part of the message should be interpreted using a Shift-JIS character set rather than as a data compaction strategy.

Note: Swiss QR Code specifies that the image shall be 46mm x 46mm (excluding quiet zone), irrespective of QR Code version, and is overlaid with a fixed 7mm x 7mm Swiss Cross pattern. The Swiss Cross pitch bears no relation to the module width of the QR symbol, so we must assume that the application provides sufficient resolution throughout the print process that defects produced by issues such as pixel grazing are insignificant. Therefore, unlike most other encoders, we do not pick a fixed module width to assist with grid fitting but rather ensure that the image is scaled to a fixed size, per the specification.

Examples

A QR-bill payment with structured creditor reference:

```
Data:
          SPC
          0200
          1
          CH5800791123000889012
          S
          Robert Schneider AG
```

```
         Rue du Lac
         1268
         2501
         Biel
         CH







         199.95
         CHF
         K
         Pia-Maria Rutschmann-Schnyder
         Grosse Marktgasse 28
         9400 Rorschach


         CH
         SCOR
         RF18539007547034

         EPD
Options:
Encoder: swissqrcode
```

5.2.15. JAB Code (Beta)

JAB Code (Just Another Barcode) is a colour 2D matrix symbology whose basic symbols are made of colourful square modules arranged in either square or rectangle grids. It was developed by Fraunhofer Institute for Secure Information Technology SIT.

Note that this is an incomplete implementation of the original BSI standard, not ISO/IEC 23634. It is effectively abandoned because there is no way to write a sufficiently fast encoder in the PostScript language due to the extremely computationally expensive LDPC error *coding used by the standard.*

Also known as: Just Another Barcode.

JAB Code uses multiple colours to encode data, achieving approximately three times higher data density compared to conventional black and white 2D matrix codes such as Data Matrix, QR Code, or Aztec Code.

Standards: BSI TR 03137 – Part 2.

Data and Options

- The data field input is the message to be encoded.
- The **parse** and **parsefnc** options enable escape sequence processing of the data field.
- The **eclevel** option specifies the error correction level from `0` to `10`, with higher values providing more error correction at the expense of data capacity. The default is `6`.
- The **colors** option specifies the number of colours to use. Valid values are `4`, `8`, `16`, `32`, `64`, `128`, or `256`. The default is `16`. More colours increase data density but require more precise printing and scanning.

Examples

A basic 16-colour symbol:

```
Data:    This is JAB Code
Options: eclevel=6
Encoder: jabcode
```

A symbol using 8 colours:

```
Data:    JAB 8 colors
Options: colors=8
Encoder: jabcode
```

5.3. One-Dimensional

5.3.1. Code 128

Code 128 is an arbitrarily long, high-density barcode symbology that can be used to encode full 256 character extended-ASCII.

Also known as: USD-6, USS-128, Code 128A, Code 128B, Code 128C.

Variants:

- GS1-128 is a variant of Code 128 that should be used when encoding data that is in GS1 Application Identifier standard format.
- HIBC Code 128 is a variant of Code 128 that should be used when encoding HIBC formatted data.

Standards: ISO/IEC 15417, ANSI/AIM BC4 - ISS Code 128, BS EN 799.

Data and Options

- The data field input can consist of any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- The mandatory check digit is calculated automatically.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII or extended-ASCII value, useful for specifying unprintable characters, e.g. `^029` for *GS*, `^209` for *Ñ*, etc.
- *FNC4* function characters will be inserted automatically to allow the encoding of extended ASCII characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1-128 for encoding GS1 AI syntax data.*
  - `^FNC2`: FNC2
  - `^FNC3`: FNC3
- *Deprecated:* The **raw** option denotes that the data field is providing the input as pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding. You can use the **alttext** option to specify human-readable text.
- *Deprecated: For internal use.* When the **parsefnc** option is specified, the special pseudo characters `^LNKA` and `^LNKC` at the end of the symbol indicate that a GS1-128 symbol includes a *CC-A/B* or *CC-C* GS1 composite 2D component.

Examples

A symbol encoding alphanumeric data:

```
Data:    Count0123456789!
Options: includetext
Encoder: code128
```

Count0123456789!

5.3.2. Code 39

The **Code 39** barcode symbology is discrete, variable length and self-checking.

Also known as: Code 3 of 9, LOGMARS, Alpha39, USD-3, USS-39.

Variants:

- Code 39 Extended is a variant of Code 39 that can be used to encode full 128 character ASCII with the use of shift character combinations.
- HIBC Code39 is a variant of Code 39 that should be used when encoding HIBC formatted data.
- AIM USD-2 is a subset of Code 39 containing the characters A-Z, 0-9, *space*, `-` and `.`.

Standards: ISO/IEC 16388, ANSI/AIM BC1 - USS Code 39, BS EN 800, MIL STD 1189.

Data and Options

- The data field can hold any of the following:
  - Numbers 0-9
  - Capital letters A-Z
  - Symbols `-.$/%*+` and *space*
- The **includecheck** option calculates the check digit.
- The **includecheckintext** option makes the calculated checksum appear in the human readable text.
- The **hidestars** option suppresses the asterisks in the human readable text.

Examples

A basic symbol:

```
Data:    CODE39
Options: includetext
Encoder: code39
```

* C O D E 3 9 *

A symbol with a check digit:

```
Data:    CODE39
Options: includecheck includetext includecheckintext
Encoder: code39
```

* C O D E 3 9 W *

A symbol with hidden start/stop characters:

```
Data:    CODE39
Options: hidestars includecheck includetext
Encoder: code39
```

C O D E 3 9

5.3.3. Code 39 Extended

The **Code 39 Extended** barcode symbology is discrete, variable length and self-checking. It is based on Code 39 but can encode full 128 character ASCII by using shift combinations.

Also known as: Code 39 Full ASCII.

Variants:

- Code 39 is a simpler variant of Code 39 Extended.

Standards: ISO/IEC 16388, ANSI/AIM BC1 - USS Code 39, BS EN 800.

Data and Options

- The data field can consist of any ASCII data.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters, e.g. `^029` for *GS*, etc.
- The **includecheck** option calculates the check digit.
- The **includecheckintext** causes the calculated check digit to appear in the human readable text.
- The **hidestars** option suppresses the asterisks in the human readable text.

Examples

A symbol encoding lowercase and special characters:

```
Data:    Code39 Ext!
Options: includetext includecheck
Encoder: code39ext
```

```
* C o d e 3 9 E x t ! *
```

A symbol using the parse option to encode a *GS* control character:

```
Data:    Code39^029Extended
Options: parse includecheck
Encoder: code39ext
```

5.3.4. Code 93

Code 93 is a continuous, variable length, self-checking barcode symbology.

Also known as: USD-7, USS-93.

Variants:

- Code 93 Extended is a variant of Code 93 that can be used to encode full 128 character ASCII with the use of special shift character combinations.

Standards: ANSI/AIM BC5 - USS Code 93, ITS 93i.

Data and Options

- The data field can hold any of the following:
  - Numbers 0-9
  - Capital letters A-Z
  - Symbols `-.$/%*+` and *space*
- The **includecheck** option calculates the two check digits.
- *Deprecated: For internal use.* The **parsefnc** option allows the special shift characters to be supplied as escape sequences:
  - `^SFT$`: ($)
  - `^SFT%`: (%)
  - `^SFT/`: (/)
  - `^SFT+`: (+)

Examples

A basic symbol:

```
Data:    CODE93
Options: includecheck
Encoder: code93
```

A symbol including a special shift combination `(/)A` representing *!*:

```
Data:    CODE93^SFT/A
Options: parsefnc includecheck
Encoder: code93
```

5.3.5. Code 93 Extended

The **Code 93 Extended** barcode symbology is continuous, variable length and self-checking. It is based on Code 93 but can encode full 128 character ASCII using four additional shift characters: *($) (%) (/) (+)*

Also known as: Code 93 Full ASCII.

Variants:

- Code 93 is a simpler variant of the Code 93 Extended barcode symbology.

Standards: ANSI/AIM BC5 - USS Code 93, ITS 93i.

Data and Options

- The data field can consist of any ASCII data.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters, e.g. `^029` for *GS*, etc.
- The **includecheck** option calculates the two check digits.

Examples

A symbol encoding lowercase and special characters:

```
Data:    Code93Ext!
Options: includecheck
Encoder: code93ext
```

A symbol using the parse option to encode a *GS* control character:

```
Data:    Code93^029Extended
Options: parse includecheck
Encoder: code93ext
```

5.3.6. Interleaved 2 of 5

Interleaved 2 of 5 is a high-density numeric barcode symbology.

Also known as: ITF, Code 2 of 5 Interleaved, USD-1, USS-Interleaved 2 of 5.

Variants:

- ITF-14 is a variant of Interleaved 2 of 5 that should be used when encoding a fourteen-digit GTIN.

Standards: ISO/IEC 16390, ANSI/AIM BC2 - USS Interleaved 2 of 5, BS EN 801.

Data and Options

- The data can consist of any number of digits.
- If the length of the symbol including the possible check digit would be odd then the data is prefixed by *0*.
- The **includecheck** option calculates the check digit.
- The **includecheckintext** option makes the calculated checksum appear in the human readable text.

Examples

A basic symbol:

```
Data:    0123456789
Options:
Encoder: interleaved2of5
```

A symbol with a check digit:

```
Data:    2401234567
Options: includecheck includetext includecheckintext
Encoder: interleaved2of5
```

0 2 4 0 1 2 3 4 5 6 7 6

5.4. Supply Chain

5.4.1. EAN-14

EAN-14 is an implementation of the GS1-128 barcode symbology with *AI (01)* that is used to encode a lone GTIN-14.

Note: EAN-14 is a colloquial term used to describe a GS1-128 barcode symbol containing only a GTIN.

Also known as: GS1-128 without attribute data.

Misnomers: UCC-14

Standards: ISO/IEC 15417, ANSI/AIM BC4-1999 ISS, BS EN 799, GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with thirteen or fourteen digits of a GTIN, i.e. `(01)....`
- Arbitrary spacing may be placed between the digits to format the human readable text without interfering with the encoded data.
- If thirteen digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)09528765432108
Options: includetext
Encoder: ean14
```

```
Data:    (01)0952876543210
Options: includetext
Encoder: ean14
```

(01)09528765432108

5.4.2. ITF-14

ITF-14 is an implementation of the Interleaved 2 of 5 barcode symbology that is typically used to encode a GTIN-14, GTIN-13 or GTIN-12.

Also known as: UPC Shipping Container Symbol, SCS, UPC Case Code.

Standards: ISO/IEC 16390, ANSI/AIM BC2-1995 USS, BS EN 801, GS1 General Specifications.

Data and Options

- The data consists of either thirteen or fourteen digits.
- Arbitrary spacing may be placed between the digits to format the human readable text without interfering with the encoded data.
- If thirteen digits are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    09528765432108
Options: includetext
Encoder: itf14
```

```
Data:    0952876543210
Options: includetext
Encoder: itf14
```

09528765432108

5.4.3. SSCC-18

SSCC-18 is an implementation of the GS1-128 barcode symbology with *AI (00)* that is typically used to encode a lone eighteen-digit shipping container serial number.

Note: SSCC-18 is a colloquial term used to describe a GS1-128 barcode symbol containing only a SSCC.

Misnomers: EAN-18, NVE.

Standards: ISO/IEC 15417, ANSI/AIM BC4-1999 ISS, BS EN 799, GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (00)* with seventeen or eighteen digits of a Serial Shipping Container Code, i.e. `(00)...`.
- Arbitrary spacing may be placed between the digits to format the human readable text without interfering with the encoded data.
- If seventeen digits of primary data are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (00)095287654321012346
Options: includetext
Encoder: sscc18
```

```
Data:    (00)09528765432101234
Options: includetext
Encoder: sscc18
```

(00)095287654321012346

5.4.4. GS1 DataMatrix

GS1 DataMatrix is an implementation of the Data Matrix (ECC 200) barcode symbology with GS1 formatted data.

Variants:

- GS1 DataMatrix Rectangular is equivalent to using `format=rectangle`.
- GS1 Digital Link Data Matrix encodes GS1 data as a Digital Link URI.

Standards: ISO/IEC 16022, ANSI/AIM BC11 ISS, GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- The **format** option is used to specify the shape of the symbol, either `square` (default) or `rectangle`.
- The **columns** and **rows** options are used to specify the size of the symbol.
- The **version** option can also be used to specify the symbol size, as `version=RxC`. Valid options are:
  - With `format=square`: *10x10*, *12x12*, *14x14*, *16x16*, *18x18*, *20x20*, *22x22*, *24x24*, *26x26*, *32x32*, *36x36*, *40x40*, *44x44*, *48x48*, *52x52*, *64x64*, *72x72*, *80x80*, *88x88*, *96x96*, *104x104*, *120x120*, *132x132*, *144x144*
  - With `format=rectangle`: *8x18*, *8x32*, *12x26*, *12x36*, *16x36*, *16x48*
- If **columns**, **rows** and **version** are unspecified the encoder will default to creating a symbol of the specified **format** that is the minimum size to represent the given data.

Examples

GTIN *9521234543213*; Weight *0.123kg*:

```
Data:    (01)09521234543213(3103)000123
```

```
Options:
Encoder: gs1datamatrix
```

5.4.5. GS1 DataMatrix Rectangular

GTIN *9521234543213*; Expiration date; Batch; Ship to location:

```
Data:    (01)09521234543213(17)120508(10)ABCD1234(410)9501101020917
Options:
Encoder: gs1datamatrixrectangular
```

5.4.6. GS1 QR Code

GS1 QR Code is an implementation of the QR Code barcode symbology with GS1 formatted data.

Variants:

- GS1 Digital Link QR Code encodes GS1 data as a Digital Link URI.

Standards: ISO/IEC 18004, ITS - QR Code, GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- The **eclevel** option is used to specify the error correction level:
  - `eclevel=L` - Low
  - `eclevel=M` - Medium (default)
  - `eclevel=Q` - Quality
  - `eclevel=H` - High
- The **fixedeclevel** option will prevent the error correction level from being

opportunistically raised when a better fix to the current matrix is possible.

- The **version** option is used to specify the size of the symbol, 1 to 40.
- If the **version** is unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data at the selected error correction level.

Examples

GTIN *9521234543213*; Extended packaging URL:

```
Data:    (01)09521234543213(8200)http://www.abc.net
Options:
Encoder: gs1qrcode
```

GTIN *9521234543213*; Extended packaging URL; Batch; Ship to location:

```
Data:    (01)09521234543213(8200)http://abc.net(10)XYZ(410)9501101020917
Options:
Encoder: gs1qrcode
```

5.4.7. GS1 Digital Link Data Matrix

GS1 Digital Link Data Matrix is an implementation of the Data Matrix (ECC 200) barcode symbology containing a GS1 Digital Link URI.

Variants:

- GS1 DataMatrix encodes GS1 data in Application Identifier format.

Standards: ISO/IEC 16022, ANSI/AIM BC11 ISS, GS1 Digital Link Standard: URI Syntax.

Data and Options

- The data field input is provided as a GS1 Digital Link URI.

- The **includetext** option enables the display of HRI.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- The **format** option is used to specify the shape of the symbol, either `square` (default) or `rectangle`.
- The **columns** and **rows** options are used to specify the size of the symbol.
- The **version** option can also be used to specify the symbol size, as `version=RxC`. Valid options are:
    - With `format=square`: *10x10*, *12x12*, *14x14*, *16x16*, *18x18*, *20x20*, *22x22*, *24x24*, *26x26*, *32x32*, *36x36*, *40x40*, *44x44*, *48x48*, *52x52*, *64x64*, *72x72*, *80x80*, *88x88*, *96x96*, *104x104*, *120x120*, *132x132*, *144x144*
    - With `format=rectangle`: *8x18*, *8x32*, *12x26*, *12x36*, *16x36*, *16x48*
- If **columns**, **rows** and **version** are unspecified the encoder will default to creating a symbol of the specified **format** that is the minimum size to represent the given data.

Examples

A GS1 Digital Link URI containing GTIN and additional data:

```
Data:    https://id.gs1.org/01/09521234543213/22/ABC%2d123?99=XYZ%2f987
Options:
Encoder: gs1dldatamatrix
```

With human readable interpretation:

```
Data:    https://id.gs1.org/01/09521234543213/22/ABC%2d123?99=XYZ%2f987
Options: includetext
Encoder: gs1dldatamatrix
```

(01)09521234543213

Note: By default, the HRI displays only the primary key extracted from the Digital Link

URI.

With full AI extraction into multiline HRI:

```
Data:    https://id.gs1.org/01/09521234543213/10/ABC123/21/12345
Options: includetext
alttext=GTIN~~~~~~~(01)~09521234543213|BATCH/LOT~(10)~ABC123|SERIAL~~~~(21)~123
45
         alttextsubspace=~ alttextsplit=| textxalign=offright textxoffset=3
         textyalign=center textfont=Courier
Encoder: gs1dldatamatrix
```

```
GTIN       (01) 09521234543213
BATCH/LOT  (10) ABC123
SERIAL     (21) 12345
```

5.4.8. GS1 Digital Link QR Code

GS1 Digital Link QR Code is an implementation of the QR Code barcode symbology containing a GS1 Digital Link URI.

Variants:

- GS1 QR Code encodes GS1 data in Application Identifier format.

Standards: ISO/IEC 18004, ITS - QR Code, GS1 Digital Link Standard: URI Syntax.

Data and Options

- The data field input is provided as a GS1 Digital Link URI.
- The **includetext** option enables the display of HRI.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- The **eclevel** option is used to specify the error correction level:
  - `eclevel=L` - Low
  - `eclevel=M` - Medium (default)
  - `eclevel=Q` - Quality
  - `eclevel=H` - High
- The **fixedeclevel** option will prevent the error correction level from being opportunistically raised when a better fix to the current matrix is possible.
- The **version** option is used to specify the size of the symbol, 1 to 40.

- If the **version** is unspecified the encoder will default to creating a symbol that is the minimum size to represent the given data at the selected error correction level.

Examples

A GS1 Digital Link URI containing GTIN and additional data:

```
Data:    https://id.gs1.org/01/09521234543213/22/ABC%2d123?99=XYZ%2f987
Options:
Encoder: gs1dlqrcode
```

With human readable interpretation:

```
Data:    https://id.gs1.org/01/09521234543213/22/ABC%2d123?99=XYZ%2f987
Options: includetext
Encoder: gs1dlqrcode
```

(01)09521234543213

Note: By default, the HRI displays only the primary key extracted from the Digital Link URI.

With full AI extraction into multiline HRI:

```
Data:    https://id.gs1.org/01/09521234543213/10/ABC123/21/12345
Options: includetext alttext=(01)~09521234543213|(10)~ABC123|(21)~12345
         alttextsubspace=~ alttextsplit=| textxalign=offright textxoffset=5
         textyalign=center
Encoder: gs1dlqrcode
```

(01) 09521234543213
(10) ABC123
(21) 12345

With a "P.o.S. ready" brand:

```
Data:    https://id.gs1.org/01/09521234543213/22/ABC%2d123?99=XYZ%2f987
Options: includetext extratext=P.o.S.~ready extratextsubspace=~
         extratextfont=Helvetica extratextsize=5 extratextyoffset=5
Encoder: gs1dlqrcode
```

P.o.S. ready

(01)09521234543213

5.4.9. GS1-128

GS1-128 is an implementation of the Code 128 barcode symbology which carries GS1 formatted data, including a GTIN-14.

Also known as: UCC/EAN-128, EAN-128, UCC-128.

Variants:

- GS1-128 Composite is a variant of GS1-128 that should be used when a CC-A, CC-B or CC-C GS1 composite 2D component is required.
- EAN-14 is a variant of GS1-128 that should be used when encoding a lone fourteen-digit GTIN.
- SSCC-18 is a variant of GS1-128 that should be used when encoding a lone eighteen-digit SSCC.

Standards: ISO/IEC 15417, ANSI/AIM BC4-1999 ISS, BS EN 799, GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.

- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- *Deprecated: For internal use.* The **linkagea** option specifies that the symbol includes a *CC-A* or *CC-B* GS1 composite 2D component.
- *Deprecated: For internal use.* The **linkagec** option specifies that the symbol includes a *CC-C* GS1 composite 2D component.

Examples

GTIN *9521234543213*; Weight *0.123kg*:

```
Data:    (01)09521234543213(3103)000123
Options: includetext
Encoder: gs1-128
```

(01)09521234543213(3103)000123

GTIN *9521234543213*; Expiration date *1st Jan 2010*; Batch *123ABC*; Serial *1234567890*:

```
Data:    (01)09521234543213(17)100101(10)123ABC(21)1234567890
Options: includetext
Encoder: gs1-128
```

(01)09521234543213(17)100101(10)123ABC(21)1234567890

5.4.10. GS1 DotCode

GS1 DotCode is an implementation of the DotCode barcode symbology with GS1 formatted data.

Variants:

- DotCode is the base symbology without GS1-specific formatting.

Standards: AIM - ISS DotCode, GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert

these automatically where necessary.

- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- The **ratio** is used to specify the aspect ratio of the symbol. The default is 1.5.
- The **columns** and **rows** options are used to specify the size of the symbol. When these are not specified a symbol is generated that maintains the selected aspect ratio.
- *Deprecated: Use only if performance is a concern.* The **fast** option is used to enable the optional high-speed symbol masking algorithm.

Examples

TPX *5vBZIF%!<B;?oa%*; GTIN *9521234543213*; Production date/time:

```
Data:    (235)5vBZIF%!<B;?oa%(01)09521234543213(8008)19052001
Options: rows=16
Encoder: gs1dotcode
```

5.5. GS1 DataBar Family

5.5.1. GS1 DataBar Omnidirectional

GS1 DataBar Omnidirectional is a fixed-length, linear barcode symbology that can be used to encode a GTIN-14 for use at point of sale.

Also known as: RSS-14.

Variants:

- GS1 DataBar Stacked Omnidirectional is a variant of GS1 DataBar Omnidirectional for use where a taller, narrower symbol is required.
- GS1 DataBar Omnidirectional Composite is a variant of GS1 DataBar Omnidirectional that should be used when a CC-A or CC-B GS1 composite 2D component is required.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with thirteen or fourteen digits of a GTIN, i.e. `(01)....`

- If thirteen digits of *AI (01)* data are supplied then the check digit is calculated automatically, otherwise the provided check digit will be verified.
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)09521234543213
Options:
Encoder: databaromni
```

```
Data:    (01)0952123454321
Options:
Encoder: databaromni
```

5.5.2. GS1 DataBar Stacked Omnidirectional

GS1 DataBar Stacked Omnidirectional is a fixed-length, stacked linear barcode symbology that can be used to encode a GTIN-14 for use a point of sale.

Also known as: RSS-14 Stacked Omnidirectional.

Variants:

- GS1 DataBar Omnidirectional is a variant of GS1 DataBar Stacked Omnidirectional for use where a shorter, wider symbol is required.
- GS1 DataBar Stacked Omnidirectional Composite is a variant of GS1 DataBar Stacked Omnidirectional that should be used when a CC-A or CC-B GS1 composite 2D component is required.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with thirteen or fourteen digits of a GTIN, i.e. `(01)....`
- If thirteen digits of *AI (01)* data are supplied then the check digit is calculated

automatically, otherwise the provided check digit will be verified.

- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)24012345678905
Options:
Encoder: databarstackedomni
```

```
Data:    (01)2401234567890
Options:
Encoder: databarstackedomni
```

5.5.3. GS1 DataBar Expanded

GS1 DataBar Expanded is a variable-length, linear barcode symbology that can be used to encode a GTIN-14 alongside a number of other application identifiers for use at point of sale.

Also known as: RSS Expanded.

Variants:

- GS1 DataBar Expanded Stacked is a variant of GS1 DataBar Expanded for use where a taller, narrower symbol is required.
- GS1 DataBar Expanded Composite is a variant of GS1 DataBar Expanded that should be used when a CC-A or CC-B GS1 composite 2D component is required.
- GS1 North American Coupon Code is an application of GS1 DataBar Expanded for use with a paperless coupon system.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- If the data contains a number of application identifiers matching any of the specifications below then they should be provided in this given order for maximum encoding efficiency:
    - `(01)9...(3103)...`
    - `(01)9...(3202)...`
    - `(01)9...(3203)...`
    - `(01)9...(310x/320x)...(11/13/15/17)...`
    - `(01)9...(310x/320x)...`
    - `(01)9...(392x)...`
    - `(01)9...(393x)...`
    - `(01)...`
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

GTIN *9521234543213*; Weight *0.123kg*:

```
Data:    (01)09521234543213(3103)000123
Options:
Encoder: databarexpanded
```

5.5.4. GS1 DataBar Expanded Stacked

GS1 DataBar Expanded Stacked is a variable-length, stacked-linear barcode symbology that can be used to encode a GTIN-14 alongside a number of other application identifiers for use at point of sale.

Also known as: RSS Expanded Stacked.

Variants:

- GS1 DataBar Expanded is a variant of GS1 DataBar Expanded Stacked for use where a shorter, wider symbol is required.
- GS1 DataBar Expanded Stacked Composite is a variant of GS1 DataBar Expanded Stacked that should be used when a CC-A or CC-B GS1 composite 2D component is required.
- GS1 North American Coupon Code is an application of GS1 DataBar Expanded for use with a paperless coupon system.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.
- *Deprecated.* The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- If the data contains a number of application identifiers matching any of the specifications below then they should be provided in this given order for maximum encoding efficiency:
  - `(01)9...(3103)...`
  - `(01)9...(3202)...`
  - `(01)9...(3203)...`
  - `(01)9...(310x/320x)...(11/13/15/17)...`
  - `(01)9...(310x/320x)...`
  - `(01)9...(392x)...`
  - `(01)9...(393x)...`
  - `(01)...`
- The **segments** option is used to specify the maximum number of segments per row which must be an even number. The default is *4*.
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

GTIN *9521234543213*; Weight *0.123kg*:

```
Data:    (01)09521234543213(3103)000123
Options: segments=4
```

```
Encoder: databarexpandedstacked
```

5.5.5. GS1 DataBar Truncated

GS1 DataBar Truncated is a fixed-length, linear barcode symbology that can be used to encode a GTIN-14 for in-house applications.

Also known as: RSS-14 Truncated.

Variants:

- GS1 DataBar Stacked is a variant of GS1 DataBar Truncated for use where a taller, narrower symbol is required.
- GS1 DataBar Truncated Composite is a variant of GS1 DataBar Truncated that should be used when a CC-A or CC-B GS1 composite 2D component is required.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with 13 or 14 digits of a GTIN, i.e. `(01)....`
- If thirteen digits of *AI (01)* data are supplied then the check digit is calculated automatically, otherwise the provided check digit will be verified.
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)09521234543213
Options:
Encoder: databartruncated
```

```
Data:    (01)0952123454321
Options:
Encoder: databartruncated
```

5.5.6. GS1 DataBar Stacked

GS1 DataBar Stacked is a fixed-length, stacked linear barcode symbology that can be used to encode a GTIN-14 for in-house applications.

Also known as: RSS-14 Stacked.

Variants:

- GS1 DataBar Truncated is a variant of GS1 DataBar Stacked for use where a shorter, wider symbol is required.
- GS1 DataBar Stacked Composite is a variant of GS1 DataBar Stacked that should be used when a CC-A or CC-B GS1 composite 2D component is required.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with thirteen or fourteen digits of a GTIN, i.e. `(01)....`
- If thirteen digits of *AI (01)* data are supplied then the check digit is calculated automatically, otherwise the provided check digit will be verified.
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)09521234543213
Options:
Encoder: databarstacked
```

```
Data:    (01)0952123454321
Options:
Encoder: databarstacked
```

5.5.7. GS1 DataBar Limited

GS1 DataBar Limited is fixed-length, linear barcode symbology that can be used to encode a GTIN-14 beginning with *0* or *1* for in-house applications.

Also known as: RSS Limited.

Variants:

- GS1 DataBar Limited Composite is a variant of GS1 DataBar Limited that should be used when a CC-A or CC-B GS1 composite 2D component is required.

Standards: ISO/IEC 24724, ITS Reduced Space Symbology (RSS), AIM ISS - Reduced Space Symbology (RSS), GS1 General Specifications.

Data and Options

- The data field input must be a solitary *AI (01)* with thirteen or fourteen digits of a GTIN starting with *0* or *1*, i.e. `(01)0...` or `(01)1...`.
- If thirteen digits of *AI (01)* data are supplied then the check digit is calculated automatically, otherwise the provided check digit will be verified.
- *Deprecated: For internal use.* The **linkage** option signifies the presence of a GS1 composite 2D component.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    (01)09521234543213
Options:
Encoder: databarlimited
```

```
Data:    (01)0952123454321
Options:
Encoder: databarlimited
```

5.5.8. GS1 North American Coupon Code

GS1 North American Coupon Code is an implementation of the GS1 DataBar Expanded barcode symbology with AI (8110) that is used as a paperless coupon system at point of sale.

Also known as: GS1 DataBar Coupon, U.S. Coupon Code.

Standards: North American Coupon Application Guideline Using GS1 DataBar Expanded Symbols, ISO/IEC 24724, GS1 General Specifications.

Data and Options

- The data field input is provided in GS1 Application Identifier standard format and must be a solitary AI (8110).
- The **segments** option is used to specify the maximum number of segments per row which must be an even number. The default is *4*.
- The following options are also relevant to this barcode symbology:
  - **coupontextfont**: PostScript font name for text above symbol
  - **coupontextsize**: Font size for the text above symbol, in points
  - **coupontextxoffset**: Horizontal position of ISBN text, in points
  - **coupontextyoffset**: Vertical position of ISBN text, in points

Examples

A coupon code with eight segments per row:

```
Data:    (8110)106141416543213500110000310123196000
Options: includetext segments=8
Encoder: gs1northamericancoupon
```

0614141-654321

5.6. Postal Symbols

5.6.1. Australia Post 4 State Customer Code

The **Australia Post 4 State Customer Code** is a barcode used by the Australian Postal Service to encode the data on letter mail.

Also known as: 4SCC, 4 State Customer Code.

Standards: Australia Post Customer Barcode Technical Specifications.

Data and Options

- The first two characters of the data field are digits used to specify the mandatory FCC type of the symbols, either:

- 11: Customer Barcode 1 ("Standard Customer Barcode")
- 45: Reply Paid Barcode
- 59: Customer Barcode 2
- 62: Customer Barcode 3
- 87: Routing Barcode (not implemented)
- 92: Redirection Barcode (not implemented)

- The next eight characters are digits that specify the mandatory DPID.
- The number of remaining characters varies according to the given FCC code and these specify the contents of the customer information field in one of two alphabets:
  - The **custinfoenc** option should be supplied as `custinfoenc=numeric` if the customer information field is to be encoded using the numeric alphabet which can contain the digits 0-9.
  - Otherwise the customer information field is encoded using the default character encoding, `custinfoenc=character`, which permits any of the following characters:
    - Upper case letters `A-Z`
    - Lower case letters `a-z`
    - Digits `0-9`
    - Symbols `space` and `#`

Examples

FCC 62 symbol with character customer data:

```
Data:    6279438541AaaB 155
Options: custinfoenc=character includetext
Encoder: auspost
```

79438541A a a B   1 5 5

FCC 59 symbol with numeric customer data:

```
Data:    5932211324012345 67
Options: custinfoenc=numeric includetext
Encoder: auspost
```

32211324012345 67

5.6.2. Deutsche Post Identcode

Deutsche Post Identcode is an implementation of the Interleaved 2 of 5 barcode symbology that is used by German Post for mail routing.

Also known as: DHL Identcode.

Standards: Deutsche Post AG Identcode.

Data and Options

- The data consists of a consecutive string of eleven or twelve digits consisting of:
  - Two-digit primary distribution centre identifier
  - Three-digit customer identifier
  - Six-digit mail piece identifier
  - One-digit check digit (may be omitted)
- If eleven digits are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    563102430313
Options: includetext
Encoder: identcode
```

```
Data:    56310243031
Options: includetext
Encoder: identcode
```

56.310 243.031 3

5.6.3. Deutsche Post Leitcode

The **Deutsche Post Leitcode** barcode symbology is an implementation of the Interleaved 2 of 5 barcode that is used by German Post for mail routing.

Also known as: DHL Leitcode.

Standards: Deutsche Post AG Leitcode.

Data and Options

- The data consists of a consecutive string of thirteen or fourteen digits consisting of:
    - Five-digit postal code
    - Three-digit street identifier
    - Three-digit house number
    - Two-digit product code
    - One-digit check digit (may be omitted)
- If thirteen digits are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    21348075016401
Options: includetext
Encoder: leitcode
```

```
Data:    2134807501640
Options: includetext
Encoder: leitcode
```

21348.075.016.40 1

5.6.4. Japan Post 4 State Barcode

The **Japan Post 4 state barcode** symbology is used by the Japan Post service to encode the delivery point identifier on letter mail.

Also known as: Kasutama Barcode, Customer Barcode.

Standards: Japan Post Customer Barcode Technical Specification.

Data and Options

- The data may contain any of the following characters:
  - Capital letters `A-Z`
  - Digits `0-9`
  - Hyphen `-`

Examples

A delivery point identifier:

```
Data:    6540123789-A-K-Z
Options:
Encoder: japanpost
```

5.6.5. MaxiCode

The **MaxiCode** barcode symbology is a 2D barcode based on a hexagonal matrix surrounding a bulls eye pattern. It can encode a structured carrier message and full 256 character extended-ASCII.

Also known as: UPS Code, Code 6, Dense Code.

Standards: ISO/IEC 16023, ANSI/AIM BC10 - ISS MaxiCode.

Data and Options

- The **mode** option is used to specify how the data is structured in the symbol:
  - `mode=2` - Formatted data containing a Structured Carrier Message with a numeric (US domestic) postal code.
  - `mode=3` - Formatted data containing a Structured Carrier Message with an alphanumeric (international) postal code.
  - `mode=4` - Unstructured extended ASCII data using standard error correction.
  - `mode=5` - Unstructured extended ASCII data using enhanced error correction.
  - `mode=6` - Barcode reader programming.
- If **mode** is unspecified the encoder will default to selecting `mode=5` if the encoded length of the input data permits enhanced error correction, otherwise it will select `mode=4` which provides standard error correction.
- The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded

using Extended Channel Interpretation (ECI).

- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
    - `^ECI000000` to `^ECI999999`: ECI indicators
- If `mode=4`, `mode=5` or `mode=6` the data field may contain any extended ASCII data.
- If `mode=2` or `mode=3` the data field must begin with a properly structured carrier message, followed by any extended ASCII data.
- The structured carrier message contains a postal code, three-digit class of service and a three-digit ISO country code separated by *GS* (ASCII 29) characters. It is formatted in the data field as follows: `[postal code]^029[country code]^029[service class]^029`. If `mode=2` the postcode must be numeric, whilst if `mode=3` the postcode may contain up to six digits, upper case letters and spaces.
- Alternatively, messages may begin with the special application field identifier "[)>{RS}01{GS}yy" where "{RS}" represents ASCII value 30, "{GS}" represents ASCII value 29 and "yy" is a two-digit year. In **parse** mode this is represented as `[)>^03001^029yy`. If `mode=2` or `mode=3` this must be immediately followed by the structured carrier message.

Examples

Identical symbols encoding unstructured data, the latter using the parse option:

```
Data:    This is MaxiCode
Options:
Encoder: maxicode
```

```
Data:    This is Maxi^067ode
Options: parse
Encoder: maxicode
```

A mode 2 symbol with a numeric postal code:

```
Data:    152382802^029840^029001^0291Z00004951^029UPSN^02906X610\
         ^029159^0291234567^0291/1^029^029Y\
         ^029634 ALPHA DR^029PITTSBURGH^029PA^029^004
Options: mode=2 parse
Encoder: maxicode
```

A mode 3 symbol with an alphanumeric postal code:

```
Data:    ABC123^029840^029001^0291Z00004951^029UPSN^02906X610\
         ^029159^0291234567^0291/1^029^029Y\
         ^029634 ALPHA DR^029PITTSBURGH^029PA^029^004
Options: mode=3 parse
Encoder: maxicode
```

A mode 2 symbol with an application field identifier:

```
Data:    [)>^03001^02996152382802^029840^029001^0291Z00004951\
         ^029UPSN^02906X610^029159^0291234567^0291/1^029^029Y\
         ^029634 ALPHA DR^029PITTSBURGH^029PA^029^004
Options: mode=2 parse
Encoder: maxicode
```

5.6.6. Royal Mail 4 State Customer Code

The **Royal Mail 4 State Customer Code** is a barcode symbology used by the British Postal Service to encode the postcode and delivery point identifier on letter mail.

Also known as: RM4SCC, CBC, BPO 4 State Code.

Standards: BS 14500.

Data and Options

- The data may contain any of the following characters from the postcode and DPID:
  - Capital letters A-Z
  - Digits 0-9
- The mandatory checksum digit is calculated automatically and must not be included in the data field

Examples

A postcode and delivery point identifier:

```
Data:    LE28HS9Z
Options: includetext
Encoder: royalmail
```

L E 2 8 H S 9 Z

5.6.7. Mailmark

Royal Mail Mailmark is an implementation of the Data Matrix (ECC 200) barcode symbology for application within the Royal Mail postal system.

Standards: Royal Mail Mailmark Barcode Definition Document.

Data and Options

- The data field input consists of 45 characters of Mailmark structured data (including required space padding) followed by variable-length, free-formatted customer data.
- Note: Due to the fixed-length field format of the message data it is important to carefully preserve spaces in the data field.
- The mandatory **type** option is used to specify the size of the symbol, either `7`, `9` or `29`.

Examples

A type 29 (144x144) symbol:

```
Data:    JGB 012100123412345678AB19XY1A 0             www.xyz.com
Options: type=29
```

```
Encoder: mailmark
```

5.6.8. Royal TNT Post 4 state barcode

The **Royal TNT Post 4 state barcode** symbology is used by the Dutch Postal Service to encode the delivery point identifier on letter mail.

Also known as: KIX, Klant IndeX.

Standards: TPG Post Kix Code Specification.

Data and Options

- The data may contain any of the following characters from the DPID:
  - Capital letters `A-Z`
  - Digits `0-9`

Examples

A delivery point identifier:

```
Data:    1231FZ13XHS
Options: includetext
Encoder: kix
```

1 2 3 1 F Z 1 3 X H S

5.6.9. USPS Intelligent Mail

The **USPS Intelligent Mail** barcode is used by the US Postal service to encode the delivery and sender information on letter mail.

Also known as: USPS OneCode.

Standards: USPS-STD-11.

Data and Options

- The data contains 31 digits representing the following:
  - Barcode Identifier - two digits
  - Service Type Identifier - three digits

    - Mailer ID, Sequence Number - either six then nine digits respectively or nine then six digits respectively
    - Delivery Point ZIP Code - eleven digits
- The mandatory checksum digit is calculated automatically and must not be included in the data field.

Examples

A 31-digit mail tracking barcode:

```
Data:    0123456709498765432101234567891
Options: includetext
Encoder: onecode
```

01 234 567094 987654321 01234 5678 91

5.6.10. USPS POSTNET

The **USPS POSTNET** barcode symbology is used by the US Postal service to encode the ZIP code information on letter mail.

Standards: USPS Publication 25.

Data and Options

- The data field contains the digits from the ZIP code, without dashes.
- The mandatory checksum is calculated automatically and must not be included in the data field.

Examples

A ZIP+4+delivery point code:

```
Data:    12345123412
Options:
Encoder: postnet
```

5.6.11. USPS PLANET

The **USPS PLANET** barcode symbology is used by the US Postal service to encode the ZIP code information on letter mail.

Standards: USPS Publication 25.

Data and Options

- The data field contains eleven or thirteen digits, without dashes.
- The mandatory checksum is calculated automatically and must not be included in the data field.

Examples

An 11-digit ZIP code:

```
Data:   01234567890
Options:
Encoder: planet
```

5.6.12. USPS FIM Symbols

The **USPS FIM** encoder is used to generate static predefined barcode symbols.

Standards: USPS Publication 25.

Data and Options

- The data field accepts one of the following values:
  - `fima` - US Postal Service FIM-A symbol
  - `fimb` - US Postal Service FIM-B symbol
  - `fimc` - US Postal Service FIM-C symbol
  - `fimd` - US Postal Service FIM-D symbol

Examples

A USPS FIM A symbol:

```
Data:    fima
Options:
Encoder: symbol
```

A USPS FIM B symbol:

```
Data:    fimb
Options:
Encoder: symbol
```

A USPS FIM C symbol:

```
Data:    fimc
Options:
Encoder: symbol
```

A USPS FIM D symbol:

```
Data:    fimd
Options:
Encoder: symbol
```

5.7. Pharmaceutical Symbols

5.7.1. Italian Pharmacode

Italian Pharmacode is a discrete, fixed length, self-checking barcode symbology used for pharmaceutical products in Italy.

Also known as: Code 32, IMH, Radix 32.

Standards: Italian Ministerial Decree 2003.

Data and Options

- The data field must contain either eight or nine digits from the code. The leading *A* which is provided in some applications must be omitted.
- The mandatory check digit is calculated automatically if it is not provided, otherwise the provided check digit is verified.

Examples

Identical symbols, input provided with and without a check digit:

```
Data:    012345676
Options: includetext
Encoder: code32
```

```
Data:    01234567
Options: includetext
Encoder: code32
```

A012345676

5.7.2. Pharmacode

Pharmacode is a binary barcode symbology that is used by the Pharmaceutical industry.

Also known as: Pharmaceutical Binary Code, Laetus Code.

Variants:

- Two-track Pharmacode is a variant of the Pharmacode barcode.

Data and Options

- The data field must contain a number between `3` and `131070` inclusive.
- The **nwidth**, **wwidth** and **swidth** options can be used to specify a custom width (in points) for the narrow bars, wide bars and inter-bar spaces respectively.

Examples

A basic symbol:

```
Data:    117480
Options:
Encoder: pharmacode
```

A symbol with custom bar widths:

```
Data:    117480
Options: nwidth=1.5 wwidth=4.5 swidth=3
Encoder: pharmacode
```

5.7.3. Two-Track Pharmacode

Two-Track Pharmacode is a binary barcode symbology used by the Pharmaceutical industry.

Also known as: Two-track Pharmaceutical Binary Code, Two-track Laetus Code.

Variants:

- Pharmacode is a variant of the Two-track Pharmacode barcode.

Data and Options

- The data field must contain a number between `4` and `64570080` inclusive.

Examples

A basic symbol:

```
Data:    117480
Options:
Encoder: pharmacode2
```

5.7.4. PZN

PZN is a discrete, fixed length, self-checking barcode symbology used for pharmaceutical products in Germany.

Also known as: Pharmazentralnummer.

Standards: IFA Coding System PZN.

Variants:

- PZN7 is the seven-digit format.
- PZN8 is the eight-digit format.

Data and Options

- For the default PZN7 encoding, the data field must contain six digits or seven digits.
- The **pzn8** option specifies that a PZN8 symbol is required, in which case the data field must contain seven digits or eight digits.
- The mandatory check digit is calculated automatically if not provided, otherwise the provided check digit is verified.
- Note: by definition, not all six-digit or seven-digit number sequences are valid inputs.

Examples

Identical PZN7 symbols, input provided with and without a check digit:

```
Data:    1234562
Options: includetext
Encoder: pzn
```

```
Data:    123456
Options: includetext
Encoder: pzn
```

PZN - 1234562

Identical PZN8 symbols, input provided with and without a check digit:

```
Data:    0275808
Options: pzn8 includetext
```

```
Encoder: pzn
```

```
Data:    02758089
Options: pzn8 includetext
Encoder: pzn
```

PZN - 02758089

5.8. HIBC Symbols

HIBC barcodes use a number of general symbologies as carrier symbols for data structured according to the LIC and PAS structured data definitions.

Variants:

- HIBC Code 39 is a variant of Code 39.
- HIBC Code 128 is a variant of Code 128.
- HIBC PDF417 is a variant of PDF417.
- HIBC MicroPDF417 is a variant of MicroPDF417.
- HIBC QR Code is a variant of QR Code.
- HIBC Data Matrix is a variant of Data Matrix.
- HIBC Data Matrix Rectangular is a variant of Data Matrix Rectangular.
- HIBC Codablock F is a variant of Codablock F.
- HIBC Aztec Code is a variant of Aztec Code.

Standards: ANSI/HIBC Provider Applications Standard, ANSI/HIBC Supplier Labelling Standard, ANSI/HIBC Positive Identification for Patient Safety, ANSI/HIBC Syntax Standard.

5.8.1. Data and Options

- The data should be pre-encoded to describe the intended barcode content.
- The HIBC + character is prefixed automatically.
- The mandatory HIBC check character is automatically appended to the input.

5.8.2. HIBC Code 39

```
Data:    A999BJC5D6E71
Options: includetext
Encoder: hibccode39
```

+A999BJC5D6E71.

5.8.3. HIBC Code 128

```
Data:    A999BJC5D6E71
Options: includetext
Encoder: hibccode128
```

+A999BJC5D6E71.

5.8.4. HIBC PDF417

```
Data:    A999BJC5D6E71
Options:
Encoder: hibcpdf417
```

5.8.5. HIBC MicroPDF417

```
Data:    A999BJC5D6E71
Options:
Encoder: hibcmicropdf417
```

5.8.6. HIBC QR Code

```
Data:    A999BJC5D6E71
Options:
Encoder: hibcqrcode
```

5.8.7. HIBC Data Matrix

```
Data:    A999BJC5D6E71
Options:
Encoder: hibcdatamatrix
```

5.8.8. HIBC Data Matrix Rectangular

```
Data:    A999BJC5D6E71
Options:
Encoder: hibcdatamatrixrectangular
```

5.8.9. HIBC Codablock F

```
Data:    A999BJC5D6E71
Options:
Encoder: hibccodablockf
```

5.8.10. HIBC Aztec Code

```
Data:    A999BJC5D6E71
```

```
Options:
Encoder: hibcazteccode
```

5.9. Less-used Symbols

5.9.1. BC412

The **BC412** barcode symbology is single width, variable length barcode that is used for silicon wafer identification by the semiconductor manufacturing industry.

Also known as: BC412 SEMI, BC412 IBM.

Standards: SEMI T1.

Data and Options

- The data field can hold any of the following:
    - Numbers 0-9
    - Capital letters A-Z, excluding O
- The **includestartstop** option enables the display of start and stop bars.
- The **includecheck** option calculates the check character.
- The **includecheckintext** option makes the calculated checksum appear in the human readable text.
- The **semi** option enables conformance to the SEMI standard by enabling start and stop bars as well as a check character.
- The **inkspread** option can be used to adjust the width of the bars.

Examples

A symbol with a check character:

```
Data:    BC412
Options: includecheck
Encoder: bc412
```

A symbol with start and stop bars:

```
Data:    BC412
Options: includestartstop
Encoder: bc412
```

A SEMI-compliant symbol:

```
Data:    BC412SEMIEXAMPLE
Options: semi
Encoder: bc412
```

5.9.2. Channel Code

Channel Code is a linear, continuous, self-checking, bidirectional barcode symbology that encodes between two and seven digits in a short space.

Standards: ANSI/AIM BC12 - USS Channel Code.

Data and Options

- The data field can hold zero prefixed values from any of the following ranges:
  - Channel 3: `00-26`
  - Channel 4: `000-292`
  - Channel 5: `0000-3493`
  - Channel 6: `00000-44072`

  - Channel 7: `000000-576688`
  - Channel 8: `0000000-7742862`
- The channel is determined to be one more than the number of digits given in the data field.
- The **shortfinder** option generates a symbol with a shortened finder pattern.
- The **includecheck** option appends an optional check bar suffix.

Examples

A channel 3 symbol holding the value five:

```
Data:    05
Options: includetext
Encoder: channelcode
```

05

A channel 4 symbol holding the value 123:

```
Data:    123
Options: includetext
Encoder: channelcode
```

123

A channel 4 symbol holding the value five including optional check bars:

```
Data:    005
Options: includetext includecheck
Encoder: channelcode
```

005

A channel 3 symbol holding the value 26 with a shortened finder pattern:

```
Data:    26
Options: shortfinder includetext
Encoder: channelcode
```

26

5.9.3. Codabar

Codabar is a linear, discrete, self-checking, bidirectional barcode symbology that can encode digits, six symbols and four delimiter characters. It is primarily used by libraries and blood banks, photo labs and FedEx airbills.

Also known as: Rationalized Codabar, Ames Code, NW-7, USD-4, USS-Codabar, ABC Codabar, Monarch, Code 2 of 7.

Standards: ANSI/AIM BC3 - USS Codabar, BS EN 798.

Data and Options

- The data field must start and stop with one of the following delimiters
  - `ABCD`
  - `TNE*` (with the *altstartstop* option)
- The data field can otherwise hold any of the following
  - Digits 0-9
  - Symbols `-$:/.+`
- The **altstartstop** option specifies that the alternative set of delimiter characters is in use.
- The **includecheck** option calculates the check digit.
- The **includecheckintext** option makes the calculated check characters appear in the human readable text.

Examples

A symbol with a check digit:

```
Data:    A0123456789B
Options: includecheck includetext includecheckintext
Encoder: rationalizedCodabar
```

A 0 1 2 3 4 5 6 7 8 9 2 B

A symbol with alternative start/stop delimiters:

```
Data:    T0123456789N
Options: altstartstop includetext
Encoder: rationalizedCodabar
```

T 0 1 2 3 4 5 6 7 8 9 N

5.9.4. Codablock F

The **Codablock F** barcode symbology is 2D stacked-linear barcode that consists of a number of stacked Code 128 symbols. It can encode full 256 character extended-ASCII.

Variants:

- HIBC Codablock F is a variant of Codablock F that should be used when encoding HIBC formatted data.

Standards: USS Codablock F.

Data and Options

- The data field can consist of any extended-ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- *FNC4* function characters will be inserted automatically to allow the encoding of extended ASCII characters.

- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^FNC3`: FNC3
- The **columns** option specifies the number of columns in the symbol, default 8.
- The **rows** option specifies the number of rows in the symbol, between 2 and 44.
- If **rows** is unspecified the encoder will default to the smallest number of rows that can hold the given data.
- The **rowheight** option specifies the height of the bars in each row in points. The default is 10.
- The **sepheight** option specifies the height of the separator bars enclosing the rows in points. The default is 1.

Examples

A basic symbol:

```
Data:    Codablock F
Options:
Encoder: codablockf
```

A symbol with fixed dimensions:

```
Data:    CODABLOCK F 34567890123456789010040digit
Options: columns=8 rows=5
Encoder: codablockf
```

A symbol with custom row and separator heights:

```
Data:    Tall bars, fat separators
Options: columns=10 rows=8 rowheight=12 sepheight=2
```

```
Encoder: codablockf
```

5.9.5. Code 11

Code 11 is a linear, discrete, non-self-checking, bidirectional, numeric barcode symbology that is primarily used for labelling telecommunication equipment.

Also known as: USD-8.

Data and Options

- The data consists of digits and the dash character -.
- The **includecheck** option calculates the check digits.
- For less than 10 data digits a single check digit is used.
- For 10 or more data digits two check digits are used.

Examples

A symbol with check digits:

```
Data:    0123456789
Options: includecheck includetext includecheckintext
Encoder: code11
```

0 1 2 3 4 5 6 7 8 9 0 3

5.9.6. Code 16K

The **Code 16K** barcode symbology is 2D stacked-linear barcode that can encode full 256 character extended-ASCII with the use of the *FNC4* shift character.

Also known as: USS-16K.

Standards: ANSI/AIM BC7 - USS Code 16K, BS EN 12323.

Data and Options

- The data field can consist of any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- *FNC4* function characters will be inserted automatically to allow the encoding of extended ASCII characters.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
    - `^FNC1`: FNC1
    - `^FNC2`: FNC2
    - `^FNC3`: FNC3
- The **sam** option specifies this symbol to be part of multi-part structured data held in up to eight symbols. For example `sam=25` specifies this to be the second symbol in a group of five symbols.
- The **rows** option specifies the number of rows in the symbol, between two and sixteen.
- If **rows** is unspecified the encoder will default to the smallest number of rows that can hold the given data.
- The **raw** option denotes that the data field is providing the input as a pre-encoded codewords in `^NNN` format, suitable for direct low-level encoding.
- The **rowheight** option specifies the height of the bars in each row in points. The default is 10.
- The **sepheight** option specifies the height of the separator bars enclosing the rows in points. The default is 1.

Examples

A basic symbol:

```
Data:    Abcd-1234567890-wxyZ
Options:
Encoder: code16k
```

A symbol with a fixed number of rows:

```
Data:    Code 16K
Options: rows=10
Encoder: code16k
```

A symbol with custom row and separator heights:

```
Data:    Tall bars, fat separators
Options: rows=8 rowheight=12 sepheight=2
Encoder: code16k
```

5.9.7. Code 25

Code 2 of 5 is a simple low density numeric barcode symbology.

Also known as: Code 25, Industrial 2 of 5, Standard 2 of 5.

Variants:

- IATA 2 of 5, Computer Identics 2 of 5.
- Datalogic 2 of 5.
- Matrix 2 of 5.
- COOP 2 of 5.

Data and Options

- The data consists of any number of digits.

- The **includecheck** option calculates the check digit.
- *Deprecated*: The **version** option determines which variant to use:
  - `version=industrial` (default) - Industrial 2 of 5.
  - `version=iata` - *Deprecated: Use IATA 2 of 5*
  - `version=datalogic` - *Deprecated: Use Datalogic 2 of 5*
  - `version=matrix` - *Deprecated: Use Matrix 2 of 5*
  - `version=coop` - *Deprecated: Use COOP 2 of 5*

Examples

A basic symbol:

```
Data:    01234567
Options:
Encoder: code2of5
```

5.9.8. DotCode

The **DotCode** symbology is 2D matrix-style barcode that can encode full 256 character extended-ASCII.

Also known as: Ribbon Code.

Variants:

- GS1 DotCode is a variant of DotCode that should be used when encoding data that is in GS1 Application Identifier standard format.

Standards: AIM - ISS DotCode.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value. This is useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified

  by escape sequences:
  - `^FNC1`: FNC1. *Recommendation: Use GS1 DotCode for encoding GS1 AI syntax data.*
  - `^FNC2`: FNC2
  - `^FNC3`: FNC3

- The **ratio** is used to specify the aspect ratio of the symbol. The default is 1.5.
- The **columns** and **rows** options are used to specify the size of the symbol. When these are not specified a symbol is generated that maintains the selected aspect ratio.
- *Deprecated: Use only if performance is a concern.* The **fast** option is used to enable the optional high-speed symbol masking algorithm.

Examples

A basic symbol:

```
Data:    This is DotCode
Options: inkspread=0.16
Encoder: dotcode
```

A symbol with custom aspect ratio:

```
Data:    Code 2.5
Options: ratio=2.5
Encoder: dotcode
```

5.9.9. Ultracode

The **Ultracode** symbology is a colour, 2D matrix-style barcode symbology that can encode full 256 character extended-ASCII.

Standards: AIM ISS - Ultracode.

Data and Options

- The data field can contain any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.

- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^FNC3`: FNC3
- The **eclevel** option is used to specify the error correction level:
  - `eclevel=EC0` - Error detection only
  - `eclevel=EC1` - Low
  - `eclevel=EC2` - Medium (default)
  - `eclevel=EC3`
  - `eclevel=EC4`
  - `eclevel=EC5` - Highest

Examples

A symbol with increased error correction:

```
Data:    Nice colours!
Options: eclevel=EC3
Encoder: ultracode
```

5.9.10. IATA 2 of 5

IATA 2 of 5 is a variant of the Code 2 of 5 barcode symbology.

Also known as: Computer Identics 2 of 5.

Variants:

- Industrial 2 of 5, Standard 2 of 5.
- Datalogic 2 of 5.
- Matrix 2 of 5.
- COOP 2 of 5.

Data and Options

- The data consists of any number of digits.

- The **includecheck** option calculates the check digit.

Examples

A symbol with a check digit:

```
Data:    01234567
Options: includetext includecheck includecheckintext
Encoder: iata2of5
```

0 1 2 3 4 5 6 7 0

5.9.11. Matrix 2 of 5

Matrix 2 of 5 is a variant of the Code 2 of 5 barcode symbology.

Variants:

- Industrial 2 of 5, Standard 2 of 5.
- IATA 2 of 5, Computer Identics 2 of 5.
- Datalogic 2 of 5.
- COOP 2 of 5.

Data and Options

- The data consists of any number of digits.
- The **includecheck** option calculates the check digit.

Examples

A symbol with a check digit:

```
Data:    01234567
Options: includetext includecheck includecheckintext
Encoder: matrix2of5
```

0 1 2 3 4 5 6 7 0

5.9.12. Datalogic 2 of 5

Datalogic 2 of 5 is a variant of the Code 2 of 5 barcode symbology.

Variants:

- Industrial 2 of 5, Standard 2 of 5.
- IATA 2 of 5, Computer Identics 2 of 5.
- Matrix 2 of 5.
- COOP 2 of 5.

Data and Options

- The data consists of any number of digits.
- The **includecheck** option calculates the check digit.

Examples

A symbol with a check digit:

```
Data:    01234567
Options: includetext includecheck includecheckintext
Encoder: datalogic2of5
```

0 1 2 3 4 5 6 7 0

5.9.13. COOP 2 of 5

COOP 2 of 5 is a variant of the Code 2 of 5 barcode symbology.

Standards: ANSI/AIM BC1 - USS Code 2 of 5.

Variants:

- Industrial 2 of 5, Standard 2 of 5.
- IATA 2 of 5, Computer Identics 2 of 5.
- Datalogic 2 of 5.
- Matrix 2 of 5.

Data and Options

- The data consists of any number of digits.
- The **includecheck** option calculates the check digit.

Examples

A symbol with a check digit:

```
Data:    01234567
Options: includetext includecheck includecheckintext
Encoder: coop2of5
```

0 1 2 3 4 5 6 7 0

5.9.14. Code 49

The **Code 49** barcode symbology is 2D stacked-linear barcode that can encode 128 character ASCII.

Also known as: USS-49.

Standards: ANSI/AIM BC6 - USS Code 49.

Data and Options

- The input can consist of any ASCII data.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^FNC2`: FNC2

  - `^FNC3`: FNC3
- The **sam** option specifies this symbol is part of multi-part structured data held in up to nine "mode 3" symbols. For example `sam=25` specifies this to be the second symbol in a group of five symbols.
- The **concat** option specifies that this symbol is an alphanumeric continuation symbol ("mode 1").
- The **rows** option specifies the number of rows in the symbol, between *2* and *8*.
- If **rows** is unspecified the encoder will default to the smallest number of rows that can hold the given data.
- The **rowheight** option specifies the height of the bars in each row in points. The default is *10*.
- The **sepheight** option specifies the height of the separator bars enclosing the rows in points. The default is *1*.

Examples

A basic symbol:

```
Data:    MULTIPLE ROWS IN CODE 49
Options:
Encoder: code49
```

A symbol with a fixed number of rows:

```
Data:    CODE 49
Options: rows=8
Encoder: code49
```

A symbol with custom row and separator heights:

```
Data:    Tall bars, fat separators
Options: rows=8 rowheight=12 sepheight=2
```

```
Encoder: code49
```

5.9.15. Code One

Code One was the earliest public domain 2D matrix-style barcode. It is used by the health care and recycling industry and can encode full 256 character extended-ASCII.

Also known as: Code 1, Code 1S.

Standards: AIM USS - Code One.

Data and Options

- The data field can consist of any ASCII data for *standard* and *T-type* symbols. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1. When supported by the receiver characters from other code pages can be encoded using Extended Channel Interpretation (ECI).
- Note: *S-type* symbols are special in that they represent a numeric value so may only contain digits.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters, e.g. `^029` for *GS*, etc.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^FNC3`: FNC3
  - `^ECI000000` to `^ECI999999`: ECI indicators
- The **version** option is used to specify the size and type of the symbol:
  - `A`, `B`, `C`, `D`, `E`, `F`, `G`, `H` - for standard format symbols (default automatic selection)
  - `version=T-16`, `version=T-32`, `version=T-48` - *T-type* symbols
  - `version=S-10`, `version=S-20`, `version=S-30` - *S-type* symbols

Examples

A standard symbol, automatically sized:

```
Data:    Code One
Options:
Encoder: codeone
```

A version C standard symbol:

```
Data:    Code One
Options: version=C
Encoder: codeone
```

A T-type symbol:

```
Data:    Code One
Options: version=T-32
Encoder: codeone
```

An S-type symbol encoding numeric data:

```
Data:    406990
Options: version=S-10
Encoder: codeone
```

5.9.16. MSI Plessey

MSI Plessey is a continuous, non-self-checking, arbitrary length, numeric barcode symbology.

Also known as: MSI, MSI Modified Plessey.

Variants:

- Plessey (UK) is the original barcode upon which MSI Modified Plessey was based.

Data and Options

- The data can consist of any number of digits.
- The **includecheck** option calculates the check digit or check digits.
- The **includecheckintext** option makes the calculated check characters appear in the human readable text.
- The **checktype** option is used to specify the type of checksum, either:
  - `checktype=mod10` (default)
  - `checktype=mod1010`
  - `checktype=mod11`
  - `checktype=ncrmod11`
  - `checktype=mod1110`
  - `checktype=ncrmod1110`
- The **badmod11** option allows a `checktype=mod11` checksum value of 10 to be encoded with a pair of check digits *10*. Normally in `checktype=mod11`, any input whose checksum evaluates to *10* is considered invalid having no correct representation.

Examples

A symbol with a mod-10 check digit:

```
Data:    0123456789
Options: includecheck includetext
Encoder: msi
```

0 1 2 3 4 5 6 7 8 9

A symbol with mod-11 then mod-10 check digits:

```
Data:    0123456789
Options: includecheck checktype=mod1110 includetext includecheckintext
```

```
Encoder: msi
```

0 1 2 3 4 5 6 7 8 9 2 9

A symbol using the badmod11 option:

```
Data:    0123456785
Options: includecheck checktype=mod11 badmod11 includetext includecheckintext
Encoder: msi
```

0 1 2 3 4 5 6 7 8 5 1 0

5.9.17. Marks & Spencer

Marks & Spencer is a barcode symbology derived from EAN-8 that is used by the UK retailer Marks & Spencer for internal purposes. It differs from EAN-8 by having the centre guard bars removed and by displaying an "M" and "S" on either side of the human readable text.

Also known as: M&S Barcode.

Data and Options

- The data field takes either seven or eight digits.
- If seven digits are supplied then the check digit is calculated automatically. Otherwise the provided check digit will be verified.
- The **includetext** option should normally be supplied.

Examples

A symbol with human-readable text:

```
Data:    06421182
Options: includetext
Encoder: mands
```

M 0642 1182 S

5.9.18. Plessey

Plessey is a continuous, arbitrary length barcode symbology for encoding hexadecimal data.

Also known as: Anker Code.

Variants:

- MSI Modified Plessey is a variant of the Plessey (UK) barcode developed by the MSI Data Corporation.

Data and Options

- The data can contain any of the following:
    - Numbers `0-9`
    - Capital letters `A-F`
- Two mandatory check characters implementing a CRC check are automatically included.
- The **includecheckintext** option makes the calculated check characters appear in the human readable text.
- The **unidirectional** option generates a unidirectional Plessey symbol.

Examples

Equivalent symbols, the latter displaying the two mandatory check characters:

```
Data:    01234ABCD
Options: includetext
Encoder: plessey
```

0 1 2 3 4 A B C D

```
Data:    01234ABCD
Options: includetext includecheckintext
Encoder: plessey
```

0 1 2 3 4 A B C D D C

A unidirectional symbol:

```
Data:    01234ABCD
Options: unidirectional
Encoder: plessey
```

5.9.19. PosiCode

PosiCode is a continuous, variable length, non-self-checking, bidirectional barcode symbology that is designed for use within printing processes where it is difficult to precisely control the width of a bar.

Standards: ITS PosiCode.

Data and Options

- The data field can hold the following:
  - For *standard* symbols: Any extended ASCII data. The default interpretation of data by readers is in accordance with ISO/IEC 8859-1.
    - *FNC4* function characters will be inserted automatically to allow the encoding of extended ASCII characters.
  - For *limited* symbols: letters `A-Z`, digits `0-9`, symbols `-` and `.`
- The **version** option is used to specify the variant of the symbol, either:
  - `version=a` (default)
  - `version=b`

- `version=limiteda`
- `version=limitedb`

- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters, e.g. `^029` for *GS*, etc.
- When the **parsefnc** option is specified, non-data function characters can be specified by escape sequences:
  - `^FNC1`: FNC1
  - `^FNC2`: FNC2
  - `^FNC3`: FNC3
- The **inkspread** option can be used to adjust the width of the bars.

Examples

PosiCode

Equivalent ways to generate a PosiCode A symbol:

```
Data:    Abc123
Options:
Encoder: posicode
```

```
Data:    Abc123
Options: version=a
Encoder: posicode
```

PosiCode A including a *GS* (ASCII 29) character:

```
Data:    AB^029CD
Options: parse
Encoder: posicode
```

PosiCode A including an *FNC2* special character:

```
Data:    AB^FNC2CD
Options: parsefnc
Encoder: posicode
```

PosiCode B symbol with widened bars:

```
Data:    Abc123
Options: version=b inkspread=-1
Encoder: posicode
```

Limited PosiCode

Limited PosiCode A:

```
Data:    ABC-12.3
Options: version=limiteda
Encoder: posicode
```

Limited PosiCode B:

```
Data:    ABC-12.3
Options: version=limitedb
Encoder: posicode
```

5.9.20. Telepen

Telepen is an arbitrary length barcode symbology for encoding all 128 ASCII characters without the need for shift characters.

Also known as: Telepen Alpha, Telepen Full ASCII.

Variants:

- Telepen Numeric.

Standards: USS Telepen.

Data and Options

- The data can contain any standard ASCII data, values 0-127.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- The mandatory check digit is automatically included.
- *Deprecated: Use Telepen Numeric instead. When the* ***numeric*** *option is given, the data is read as either pairs of digits or* `0X`*,* `1X`*, etc. The singular values* `^000` *to* `^016` *can also be encoded using the _parse* option._

Examples

A basic symbol:

```
Data:    Telepen
Options: includetext
Encoder: telepen
```

T e l e p e n

A symbol using the parse option to encode a carriage return:

```
Data:    Telepen^013
Options: parse
Encoder: telepen
```

5.9.21. Telepen Numeric

Telepen Numeric is a variant of the Telepen symbology for efficient encoding of numeric data.

Variants:

- Telepen Alpha.

Standards: USS Telepen.

Data and Options

- The data is provided as either pairs of digits or `0X`, `1X`, etc. The singular values `^000` to `^016` can also be encoded using the *parse* option.
- When the **parse** option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- The mandatory check digit is automatically included.

Examples

A symbol encoding pairs of digits:

```
Data:    123456
Options:
Encoder: telepennumeric
```

A symbol encoding data that includes X digits:

```
Data:    1X345X
Options:
Encoder: telepennumeric
```

5.10. GS1 Composite Symbols

GS1 Composite barcode symbologies consist of a primary component beneath a 2D component (variations of MicroPDF417 and PDF417) used to encode supplementary GS1 formatted data.

Variants:

- EAN-13 Composite is a variant of EAN-13.
- EAN-8 Composite is a variant of EAN-8.
- UPC-A Composite is a variant of UPC-A.
- UPC-E Composite is a variant of UPC-E.
- GS1 DataBar Omnidirectional Composite is a variant of GS1 DataBar Omnidirectional.
- GS1 DataBar Stacked Omnidirectional Composite is a variant of GS1 DataBar Stacked Omnidirectional.
- GS1 DataBar Expanded Composite is a variant of GS1 DataBar Expanded.
- GS1 DataBar Expanded Stacked Composite is a variant of GS1 DataBar Expanded Stacked.
- GS1 DataBar Truncated Composite is a variant of GS1 DataBar Truncated.
- GS1 DataBar Stacked Composite is a variant of GS1 DataBar Stacked.
- GS1 DataBar Limited Composite is a variant of GS1 DataBar Limited.
- GS1-128 Composite is a variant of GS1-128.

Standards: ISO/IEC 24723, ITS EAN.UCC Composite Symbology, AIM ISS - EAN.UCC Composite Symbology, GS1 General Specifications.

5.10.1. Data and Options

- The data field consists of a primary and secondary component separated by a pipe `|` character.
- The data for the primary component (preceding the pipe) is entered in a format identical to the corresponding non-composite barcode symbology.
- The data for the 2D component (following the pipe) is entered in GS1 Application Identifier standard format.
- Note that it is not necessary to specify *FNC1* characters since the encoder will insert these automatically where necessary.
- *Deprecated*. The **dontlint** option disables GS1 Application Identifier syntax validation allowing images to be generated for input that does not comply with GS1 standards.
- For maximum efficiency, if the data for the 2D component contains a number of application identifiers matching any of the specifications below then they should be provided in this given order:
    - `(11)...(10)...`
    - `(17)...(10)...`
    - `(90){0-3 digits not starting 0}{upper alpha}...`
- *Deprecated: Default is best unless your application dictates otherwise.* The **ccversion** option is used to select a specific 2D component:
    - `ccversion=a` - CC-A
    - `ccversion=b` - CC-B
    - `ccversion=c` - CC-C (GS1-128 Composite only)
    - If **ccversion** is not specified a CC-A component will be selected if the data will fit, otherwise a CC-B component will be used. In the case of GS1-128 Composite a CC-C component will be used if the data does not fit within either a CC-A or CC-B component.

5.10.2. EAN-13 Composite

```
Data:    9520123456788|(99)1234-abcd
Options: includetext guardwhitespace
Encoder: ean13composite
```

9 520123 456788 >

5.10.3. EAN-8 Composite

```
Data:    95200002|(21)A12345678
Options: includetext guardwhitespace
Encoder: ean8composite
```

< 9520 0002 >

5.10.4. UPC-A Composite

```
Data:    012345000058|(99)1234-abcd
Options: includetext
Encoder: upcacomposite
```

0 12345 00005 8

5.10.5. UPC-E Composite

```
Data:    01234558|(15)021231
Options: includetext
Encoder: upcecomposite
```

0 123455 8

5.10.6. GS1 DataBar Omnidirectional Composite

```
Data:    (01)09521234543213|(11)990102
Options:
Encoder: databaromnicomposite
```

5.10.7. GS1 DataBar Stacked Omnidirectional Composite

```
Data:    (01)09521234543213|(11)990102
Options:
Encoder: databarstackedomnicomposite
```

5.10.8. GS1 DataBar Expanded Composite

```
Data:    (01)09521234543213(3103)001234|(91)1A2B3C4D5E
Options:
Encoder: databarexpandedcomposite
```

5.10.9. GS1 DataBar Expanded Stacked Composite

```
Data:    (01)09521234543213(10)ABCDEF|(21)12345678
```

```
Options: segments=4
Encoder: databarexpandedstackedcomposite
```

5.10.10. GS1 DataBar Truncated Composite

```
Data:    (01)09521234543213|(11)990102
Options:
Encoder: databartruncatedcomposite
```

5.10.11. GS1 DataBar Stacked Composite

```
Data:    (01)09521234543213|(17)010200
Options:
Encoder: databarstackedcomposite
```

5.10.12. GS1 DataBar Limited Composite

```
Data:    (01)09521234543213|(21)abcdefghijklmnopqrst
Options:
Encoder: databarlimitedcomposite
```

5.10.13. GS1-128 Composite

GS1-128 Composite with a CC-A 2D component:

```
Data:    (01)09521234543213|(21)A1B2C3D4E5F6G7H8
Options:
Encoder: gs1-128composite
```

GS1-128 Composite with a CC-C 2D component:

```
Data:    (00)095287654321012346|(02)09521234543213(37)24(10)1234567ABCDEFG
Options: ccversion=c
Encoder: gs1-128composite
```

5.10.14. CC-A

Isolated CC-A 2D component:

```
Data:    (01)09521234543213
Options: ccversion=a cccolumns=3
Encoder: gs1-cc
```

5.10.15. CC-B

Isolated CC-B 2D component:

```
Data:    (01)09521234543213(3103)000123
Options: ccversion=b cccolumns=4
Encoder: gs1-cc
```

5.10.16. CC-C

Isolated CC-C 2D component:

```
Data:    (02)09521234543213(37)24(10)1234567ABCDEFG
Options: ccversion=c cccolumns=5
Encoder: gs1-cc
```

5.11. Raw Symbols

5.11.1. DAFT

DAFT is an encoder for directly specifying the descender, ascender, full-height, tracker-bar succession for a custom 4 state barcode symbol.

Data and Options

- The data field contains a sequence of the characters `D`, `A`, `F` or `T` to denote the descender, ascender, full-height and tracker bars of a custom 4 state symbol.

Examples

A custom 4-state symbol:

```
Data:    FATDAFTDAD
Options:
Encoder: daft
```

5.11.2. Flattermarken

Flattermarken are identification marks used in book production that facilitate the proper arrangement of bound sections by a book binder.

Data and Options

- The data field holds a sequence of digits corresponding to a 9 module width with the following meaning:
  - `1-9`: a single mark exists in the corresponding module position
  - `0`: unmarked sequence of modules
- The **inkspread** option can be used to adjust the width of the bars.
- If greater fidelity is required then the raw encoder should be used instead.

Examples

Marks at positions 1, 3 and 4:

```
Data:    1304
Options: inkspread=-1
Encoder: flattermarken
```

5.11.3. Raw

The **raw** encoder is used for directly specifying the space/bar succession of a custom barcode symbol.

Data and Options

- The data field contains an alternating sequence of widths (1 to 9) for the bars and spaces of a custom symbol.

Examples

A custom barcode specifying bar/space widths directly:

```
Data:    331132131313411122131311333213114131131221323
Options: height=0.5
Encoder: raw
```

5.12. Partial Symbols

5.12.1. EAN-2

EAN-2 is the two-digit add-on code that accompanies a EAN-13 or UPC type barcode symbol such as an ISBN or ISSN.

Also known as: Two-Digit Add-On, Two-Digit Supplement, UPC-2.

Standards: ISO/IEC 15420, GS1 General Specifications.

Note: The use of a two-digit add-on with EAN-8 is not defined by any symbology standard.

Deprecated. Standalone use of the EAN-2 encoder is discouraged. It is better to include an addon by specifying the supplicant's value as an extension to the data for the primary symbol. For example, `0317-8471 03 17` for a symbol representing the ISSN 0317-8471 (having sequence number 03) and with issue number 17 denoted using a two-digit addon.

Data and Options

- The data field must contain two digits.
- The **includetext** option should normally be supplied.

Examples

A standalone two-digit add-on:

```
Data:    05
Options: includetext guardwhitespace
Encoder: ean2
```

5.12.2. EAN-5

EAN-5 is the five-digit add-on code that accompanies an EAN or UPC type barcode symbol such as an ISBN or ISSN.

Also known as: Five-Digit Add-On, Five-Digit Supplement, UPC-5.

Standards: ISO/IEC 15420, GS1 General Specifications.

Note: The use of a five-digit add-on with EAN-8 is not defined by any symbology standard.

Deprecated. Standalone use of the EAN-5 encoder is discouraged. It is better to include an addon by specifying the supplicant's value as an extension to the data for the primary symbol. For example, `978-1-873671-00-9 54499` for a symbol representing the ISBN 978-1-873671-00-9 with recommended retail price $44.99 USD denoted using a five-digit addon.

Data and Options

- The data field must contain five digits.
- The **includetext** option should normally be supplied.

Examples

A standalone five-digit add-on:

```
Data:    90200
Options: includetext guardwhitespace
Encoder: ean5
```

90200>

5.13. GS1 Application Identifier Standard Format

Certain barcode symbologies (including GS1-128, GS1 DataBar Expanded, GS1 DataMatrix, GS1 QR Code and GS1 Composite Symbols) represent standardized GS1 data and require that their data field is provided in GS1 Application Identifier standard format, consisting of a concatenated string of *AIs* along with their corresponding values.

The AIs are a set of approximately one hundred two-, three- or four-digit prefixes written within parentheses that represent physical attributes and business information, e.g.

- *(00)* is an eighteen-digit SSCC.
- *(01)* is a fourteen-digit GTIN.
- *(403)* is a variable-length routing code.

The following input represents GTIN-14 *09521234543213*; Expiration Date *1 January 2010*; Batch *123ABC*; Serial *1234567890*:

```
(01)09521234543213(17)100101(10)123ABC(21)1234567890
```

Encoders for barcode symbologies that expect data in GS1 Application Identifier standard format will take care of parsing the input and inserting any necessary *FNC1* characters to delimit variable length fields.

```
Input to GS1-based encoder:
(01)09521234543213(17)100101(10)123ABC(21)1234567890
                                                |
                                [ BWIPP encodes barcode message... ]
                                                |
                                                V
Barcode contents:
```

```
{FNC1}0952123454321317100101110123ABC{FNC1}211234567890
                                                       |
                                   [ Scanner transfer protocol sends... ]
                                                       |
                                                       V
Application receives:
]Nm0952123454321317100101110123ABC{GS}211234567890

(Where "]Nm" is the symbology specific AIM Symbology Identifier.)
```

5.13.1. Encoding parentheses within Application Identifier data values

Instances of parentheses, `(` and `)`, within Application Identifier data values must be escaped using the parse option as `^040` and `^041`, respectively.

5.13.2. GS1 Application Identifier Linting

The input is checked against the structural rules for the GS1 Application Identifiers, as described in this article, unless the **dontlint** option is given. The Application Identifier definitions are provided in the GS1 General Specifications.

5.13.3. Use of Group Separator character (GS; ASCII 29) as an AI separator

FNC1 is the canonical separator that is used in GS1 symbols when encoding Application Identifier based data.

The transfer protocol for FNC1 characters is defined by the fundamental ISO/IEC barcode symbology standards and is not something that an application standard can legitimately redefine whilst claiming conformance with the fundamental standards.

Encoding a GS (ASCII value 29) directly within a barcode message is incorrect and may lead to larger symbol sizes since they may not be encoded as efficiently and FNC1. However, because the transfer protocol for barcode scanners converts FNC1 non-data characters in third and subsequent characters positions to GS data characters for transmission, the host is generally unable to determine whether FNC1 or GS was used in the barcode message.

Because of this ambiguity, the use of GS characters of AI separation is tolerated by the GS1 standards, but not encouraged.

BWIPP will always correctly encode the barcode message using FNC1 characters.

If you wish to do the wrong thing then you can manually encode the data and use a general (non-GS1) instance of the encoder, for example:

Instead of auto-encoding with canonical use of FNC1 separators:

```
Encoder:   GS1 DataMatrix
Data:      (01)09521234543213(17)100101(10)123ABC(21)1234567890
Options:
```

Manually encode with the non-canonical use of GS separators:

```
Encoder:   Data Matrix
Data:      ^FNC1095212345432131710010110123ABC^GS211234567890
Options:   parse parsefnc
```

Note however that:

- The rules for encoding Application Identifier based data are nuanced. FNC1/GS separators are not merely separators and their necessity is unrelated to whether the AIs are fixed or variable length, as many sources falsely claim.
- You will not benefit from BWIPP's comprehensive validation of AI-based data if you choose to encode data manually rather than let BWIPP take care of it.

Chapter 6. Options Reference

6.1. Check Digits

6.1.1. includecheck

Generate check digit(s) for symbologies where the use of check digits is optional.

Example

Calculate the optional check characters of this Code 93 symbol:

```
Data:    CHECK ME OUT
Options: includecheck
Encoder: code93
```

6.1.2. includecheckintext

Show the calculated check digit in the human readable text.

Notes

- For barcode symbologies where the check digit is not mandatory, this option must be used in combination with **includecheck**.
- If any part of the checksum does not have a printable representation then that part is not displayed.

Example

Display the check digit of this Royal Mail barcode:

```
Data:    LE28HS9Z
Options: includetext includecheckintext
Encoder: royalmail
```

L E 2 8 H S 9 Z 9

6.2. Input Processing

6.2.1. parse

In supporting barcode symbologies, when the *parse* option is specified, any instances of `^NNN` in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.

Additionally, control character names can be used to specify equivalent ASCII values, as follows:

Sequence	ASCII value
`^NUL`	0
`^SOH`	1
`^STX`	2
`^ETX`	3
`^EOT`	4
`^ENQ`	5
`^ACK`	6
`^BEL`	7

Sequence	ASCII value
`^BS`	8
`^TAB`	9
`^LF`	10
`^VT`	11
`^FF`	12
`^CR`	13
`^DLE`	16
`^DC1`	17
`^DC2`	18
`^DC3`	19
`^DC4`	20
`^NAK`	21
`^SYN`	22
`^ETB`	23
`^CAN`	24
`^EM`	25
`^SUB`	26
`^ESC`	27
`^FS`	28
`^GS`	29
`^RS`	30
`^US`	31

ASCII control characters `SO` and `SI` cannot be encoded by name (since `SO` would prefix clash with `SOH`).

Note: When this option is enabled, literal instances of "^" in the data should be escaped as `^094` to avoid replacement if the subsequent data characters represent a valid substitution. For example, literal "^123" in the input data can be escaped as `^094123`, literal "^RS" in the data can be escaped as `^094RS`, and literal "^ABC" in the data can be escaped as `^094ABC` (in case `^ABC` becomes a valid substitution in the future).

Example

Equivalent symbols:

```
Data:    This is Data Matrix
Options:
Encoder: datamatrix
```

```
Data:    This is ^068ata Matrix
Options: parse
Encoder: datamatrix
```

6.2.2. parsefnc

In supporting barcode symbologies, when the **parsefnc** option is specified, non-data function characters can be specified by escaped combinations such as `^FNC1`, `^FNC4`, `^SFT/` and `^ECI000003`.

Note: When this option is enabled, literal instances of `^` in the input data can be escaped as `^^`.

Example

Manually-composed Code 93 Extended demonstrating use of the special shift combination `(/)A` to represent *!*:

```
Data:    TERRY^SFT/A
Options: parsefnc includecheck
Encoder: code93
```

6.3. Symbol Dimensions

6.3.1. height

Height of longest bar, in inches.

Example

A 1/2 inch tall EAN-13:

```
Data:    9520123456788
Options: includetext height=0.5
Encoder: ean13
```

9 520123 456788

6.3.2. width

Stretch the symbol to precisely this width, in inches.

Notes

- This parameter literally stretches the symbol and text to the desired width which may distort the human readable text.
- For information about resizing symbols read the article on resizing symbols.

Example

A 2 inch wide Code 93 symbol:

```
Data:    TERRY
Options: width=2
Encoder: code93
```

6.4. Element Properties

These are options that allow you to compensate for print gain or print loss resulting from the properties of the printing process or substrate.

6.4.1. inkspread

For linear barcodes, the amount by which to reduce the bar widths to compensate for inkspread, in points.

For matrix barcodes, the amount by which to reduce the width and height of dark modules to compensate for inkspread, in points.

For MaxiCode, the amount by which to reduce the width and height of hexagons to compensate for inkspread, in points.

Notes

- Negative values will increase the bar width.

Example

```
Data:    TEZ
Options: inkspread=0.6
Encoder: code39
```

6.4.2. inkspreadh

For matrix barcodes, the amount by which to reduce the width of dark modules to compensate for inkspread, in points.

Note: inkspreadh is most useful for stacked-linear type barcodes such as PDF417 and Codablock F.

Example

```
Data:    Aztec
Options: inkspreadh=0.5
Encoder: azteccode
```

6.4.3. inkspreadv

For matrix barcodes, the amount by which to reduce the height of dark modules to compensate for inkspread, in points.

Example

```
Data:    Test
Options: inkspreadv=0.5
Encoder: ultracode
```

6.4.4. dotty

For matrix barcodes, render the modules as dots rather than squares. The dot radius can be adjusted using the **inkspread** option.

Example

```
Data:    DOTTY
Options: dotty inkspread=-0.2 height=2 width=2
Encoder: datamatrix
```

6.5. Text Properties

6.5.1. includetext

Show human readable text for data in symbol.

Notes

- If a character in the data does not have a printable representation then it is not displayed

Example

```
Data:    SEE ME
Options: includetext
Encoder: code39
```

6.5.2. textfont

The PostScript font name for text.

Notes

- The font name must be the literal name of a PostScript font that is available to the system.
- This option should be used in combination with the **includetext** option.
- The PostScript font name for a font file can be determined with the following command from the fontconfig library:
  - `fc-scan --format "%{postscriptname}\n" /usr/share/fonts/truetype/somefont.ttf`

Example

```
Data:    FONT
Options: includetext textfont=Times-Roman
Encoder: code39
```

6.5.3. textsize

The font size of the text in points.

Note

- This option should be used in combination with the **includetext** option.

Example

```
Data:    SIZE
Options: includetext textsize=20 textyoffset=-12
Encoder: code39
```

SIZE

6.5.4. textgaps

The inter-character spacing of the text.

Note

- This option should be used in combination with the **textxalign** option.

Example

```
Data:    HELLO
Options: includetext textxalign=center textgaps=10
Encoder: code128
```

H E L L O

6.5.5. alttext

Specify text to display other than what is provided in the data field.

Note

- This option should be used in combination with the **includetext** option.

Example

```
Data:    ABC123
Options: includetext alttext=CUSTOM-TEXT
Encoder: code128
```

CUSTOM-TEXT

6.5.6. alttextsubspace

Specify a set of characters that will be replaced with spaces within the **alttext** value.

Note

- This option should be used in combination with the **alttext** option.

Example

```
Data:    ABC123
Options: includetext alttext=WITH~SPACES alttextsubspace=~
Encoder: code128
```

WITH SPACES

6.5.7. alttextsplit

Specify a character by which to split **alttext** into separate lines.

Note

- This option should be used in combination with the **alttext** option.

Example

```
Data:    ABC123
Options: includetext alttext=LINE~ONE|LINE~TWO alttextsubspace=~
```

```
alttextsplit=|
Encoder: code128
```

LINE ONE
LINE TWO

6.5.8. textlinegaps

Specify the gap between consecutive text lines.

Note

- This option should be used in combination with the **alttextsplit** option.

Example

```
Data:    ABC123
Options: includetext alttext=LINE~ONE|LINE~TWO alttextsubspace=~
         alttextsplit=| textlinegaps=10
Encoder: code128
```

LINE ONE

LINE TWO

6.5.9. extratext

Extra text can be rendered above and to the left of the symbol (default) by setting `extratext` option to it.

Example

```
Data:    Hello
Options: version=8 includetext alttext=Hello extratext=SCAN~ME
         extratextsubspace=~ extratextfont=Helvetica-Bold
         extratextsize=8 extratextyoffset=6
Encoder: qrcode
```

SCAN ME

Hello

6.5.10. extratextfont, extratextsize, extratextgaps

The formatting of extra text can be specified using the `extratextfont`, `extratextsize` and `extratextgaps` options, which function the same as their `textfont`, `textsize` and `textgaps` analogues.

6.5.11. extratextsubspace, extratextsplit, extratextlinegaps

Extra text supports the same substitution and line splitting options as alttext via `extratextsubspace`, `extratextsplit` and `extratextlinegaps`.

6.6. Text Positioning

6.6.1. textxalign

The **textxalign** option is used to specify where to horizontally position the text.

Example: textxalign=offleft

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offleft
Encoder: azteccode
```

Test

Example: textxalign=left

```
Data:    Test
```

```
Options: includetext alttext=Test layers=8 textxalign=left
Encoder: azteccode
```

Test

Example: textxalign=center

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=center
Encoder: azteccode
```

Test

Example: textxalign=right

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=right
Encoder: azteccode
```

Test

Example: textxalign=offright

```
Data:    Test
```

```
Options: includetext alttext=Test layers=8 textxalign=offright
Encoder: azteccode
```

Test

6.6.2. textyalign

The **textyalign** option is used to specify where to vertically position the text.

Example: textyalign=below

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
textyalign=below
Encoder: azteccode
```

Test

Example: textyalign=bottom

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
textyalign=bottom
Encoder: azteccode
```

Test

Example: textyalign=center

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
textyalign=center
Encoder: azteccode
```

Test

Example: textyalign=top

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright textyalign=top
Encoder: azteccode
```

Test

Example: textyalign=above

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
textyalign=above
Encoder: azteccode
```

Test

6.6.3. textdirection

The **textdirection** option is used to specify in which direction to orient text.

Example: textdirection=forward

```
Data:    Test
Options: includetext alttext=Test layers=8 textdirection=forward
Encoder: azteccode
```

Example: textdirection=backward

```
Data:    Test
Options: includetext alttext=Test layers=8 textdirection=backward
Encoder: azteccode
```

Example: textdirection=upward

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
         textyalign=center textdirection=upward
Encoder: azteccode
```

Test

Example: textdirection=downward

```
Data:    Test
Options: includetext alttext=Test layers=8 textxalign=offright
         textyalign=center textdirection=downward
Encoder: azteccode
```

Test

6.6.4. textxoffset

The horizontal position of the text in points relative to the leftmost bar or modules.

Example

```
Data:    TEST
Options: includetext textxalign=center textxoffset=-40
Encoder: code39
```

TEST

6.6.5. textyoffset

The vertical position of the text in points relative to the base of the bars or modules.

Example

```
Data:    TEST
Options: includetext textyoffset=-20
Encoder: code39
```

* T E S T *

Notes

- By default (in the absence of **textxalign** or **textyalign**), each character of text is placed immediately below the corresponding modules where this is possible.
- Where there isn't such a direct relationship then the default is to position the text centrally beneath the symbol.

6.6.6. extratextxalign, extratextyalign, extratextdirection, extratextxoffset, extratextyoffset

Any extra text can be positioned using the `extratextxalign`, `extratextyalign`, `extratextdirection`, `extratextxoffset` and `extratextyoffset` options that function the same as their `textxalign`, `textyalign`, `textdirection`, `textxoffset` and `textyoffset` analogues.

6.7. Border Properties

6.7.1. showborder

Display a border around the graphical (non-text) components of the symbol, e.g. bar/space array, matrix, composite separator. By default the border (and any background) will designate the symbol's quiet zone.

Example

Display a customised border around this Codabar symbol:

```
Data:    A3+14-63B
Options: includetext showborder borderwidth=2 borderbottom=8
Encoder: rationalizedCodabar
```

6.7.2. showbearer

Display bearer bars above and below the graphical (non-text) components of the symbol, e.g. bar/space array, matrix, composite separator. By default the bearer bars will designate the symbol’s quiet zone.

Example

Interleaved 2 of 5 with bearer bars:

```
Data:    24012345678905
Options: includetext showbearer borderwidth=3
Encoder: interleaved2of5
```

6.7.3. borderwidth

Width of the border/bearers, in points.

6.7.4. borderleft

Custom gap between the left edge of the symbol and the border or bearer bars, in points.

Example

```
Data:    Border
Options: showborder borderwidth=1 borderleft=10
Encoder: datamatrix
```

6.7.5. borderright

Custom gap between the right edge of the symbol and the border or bearer bars, in points.

Example

```
Data:    Border
Options: showborder borderwidth=1 borderright=10
Encoder: datamatrix
```

6.7.6. bordertop

Custom gap between the top edge of the symbol and the border or bearer bar, in points.

Example

```
Data:    Border
Options: showborder borderwidth=1 bordertop=10
Encoder: datamatrix
```

6.7.7. borderbottom

Custom gap between the bottom edge of the symbol and the border or bearer bar, in points.

Example

```
Data:    Border
Options: showborder borderwidth=1 borderbottom=10
Encoder: datamatrix
```

6.8. Symbol Colors

Colors can be specified as a hex `RRGGBB` value (RGB) or a hex `CCMMYYKK` value (CMYK) or a predefined color name beginning with "_".

For information about advanced use of colors (separations, non-standard colour spaces, etc.) read the article on named colors.

6.8.1. barcolor

Color of the bars or dark modules.

Example

```
Data:    Color
Options: layers=6 barcolor=AA0000
Encoder: azteccode
```

6.8.2. backgroundcolor

Color of the light background or light modules.

Example

```
Data:    Color
Options: layers=6 backgroundcolor=FFFFCC
Encoder: azteccode
```

6.8.3. bordercolor

Color of the border.

Example

```
Data:    Color
Options: layers=6 showborder borderwidth=3 bordercolor=0000AA
Encoder: azteccode
```

6.8.4. textcolor

Color of the text.

Example

```
Data:    Color
Options: layers=6 includetext alttext=Color textcolor=008800
Encoder: azteccode
```

Color

6.8.5. extratextcolor

Color of any extra text.

Example

```
Data:    Color
Options: layers=6 extratext=LABEL extratextcolor=AA00AA
Encoder: azteccode
```

LABEL

6.8.6. Combined Example

Multiple color options used together:

```
Data:    Color
Options: layers=6 includetext alttext=Color textyoffset=-3
         extratext=LABEL extratextyoffset=4 showborder borderwidth=2
         barcolor=2E5090 backgroundcolor=FFF4E6 bordercolor=5A8F7B
         textcolor=4A4A4A extratextcolor=C85A2E
Encoder: azteccode
```

LABEL

Color

6.9. EAN-UPC Add-ons

These options control the appearance of the add-on (supplement) text for EAN/UPC barcodes with 2-digit or 5-digit add-ons.

6.9.1. addongap

The gap between the main barcode and the add-on, in points. Default is 12.

Example

```
Data:    5012345678900 12
Options: includetext addongap=20
Encoder: ean13
```

12

5 012345 678900

6.9.2. addontextfont

The font name of the add-on text.

Note

- The font name must be the literal name of a PostScript available font.

Example

```
Data:    5012345678900 12
Options: includetext addontextfont=Courier
Encoder: ean13
```

12

5 012345 678900

6.9.3. addontextsize

The size of the add-on text, in points.

Example

```
Data:    5012345678900 12
Options: includetext addontextsize=15
Encoder: ean13
```

12

5 012345 678900

6.9.4. addontextxoffset

Overrides the default horizontal positioning of the add-on text.

Example

```
Data:    5012345678900 12
Options: includetext addontextxoffset=-10
Encoder: ean13
```

6.9.5. addontextyoffset

Overrides the default vertical positioning of the add-on text.

Example

```
Data:    5012345678900 12
Options: includetext addontextyoffset=-10
Encoder: ean13
```

6.10. EAN-UPC Guards

These options control the whitespace guards (quiet zone indicators) for EAN/UPC barcodes.

6.10.1. guardwhitespace

Display whitespace guards.

Example

```
Data:    95200002
Options: includetext guardwhitespace
Encoder: ean8
```

6.10.2. guardwidth

Width of the whitespace guards, in points.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardwidth=0
Encoder: ean8
```

6.10.3. guardheight

Height of the whitespace guards, in points.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardheight=15
Encoder: ean8
```

6.10.4. guardleftpos

Override the default horizontal position of the left whitespace guard.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardleftpos=10
Encoder: ean8
```

< 9520 0002 >

6.10.5. guardrightpos

Override the default horizontal position of the right whitespace guard.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardrightpos=10
Encoder: ean8
```

< 9520 0002 >

6.10.6. guardleftypos

Override the default vertical position of the left whitespace guard.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardleftypos=20
Encoder: ean8
```

6.10.7. guardrightypos

Override the default vertical position of the right whitespace guard.

Example

```
Data:    95200002
Options: includetext guardwhitespace guardrightypos=20
Encoder: ean8
```